UNIVERSITÀ DEGLI STUDI DI TORINO

Polo delle scienze della natura

Corso di Laurea in Informatica

A VISUAL AUDITORY MODEL BASED ON GROWING SELF-ORGANIZING MAPS TO ANALYZE THE TAXONOMIC RESPONSE IN EARLY CHILDHOOD

Tesi di Laurea Magistrale

Anno Accademico 2014/2015

Relatrice:
Valentina Gliozzi

Candidato:
Matteo Madeddu

I am inclined to doubt that anything very resembling formal logic could be a good model for human reasoning.

Marvin Lee Minsky

RINGRAZIAMENTI

Un sentito grazie va a Valentina Gliozzi, per tutto il suo tempo, la pazienza e i caffè offerti, ma soprattutto per aver saputo ascoltare e consigliarmi anche al di fuori del contesto di lavoro.

Vorrei ringraziare i miei genitori, per avermi trasmesso tutti gli insegnamenti e i valori che contano di più, e che non si possono apprendere sui libri. Mia sorella, Alice, perché con il suo ottimismo e la sua tenacia, è e continua ad essere un modello a cui ispirarsi.

Valentina, semplicemente insostituibile, in ogni istante paziente e gentile, soprattutto nei momenti più difficili. Grazie di esistere e rimanere sempre al mio fianco.

Vorrei ringraziare Matteo e Chiara, a partire dalle piccole cose, per ogni messaggio, per ogni uscita settimanale, per ogni minuto in cui mi hanno ascoltato, per continuare ad essere due amici insostituibili, anche nei periodi di lontananza.

Vorrei ringraziare chi non c'è più, perché mi piace pensare che anche i vostri insegnamenti abbiano contribuito a questo lavoro.

THANKS

A heartfelt thanks goes to Valentina Gliozzi, for all her time, patience, and coffees, but above all for being able to listen and advise me even outside of the work context.

I would like to thank my parents, for conveying to me all the teachings and the values that matter most, and I could not learn from books. My sister, Alice, because with her optimism and tenacity, is and continues to be a role model.

Valentina, simply irreplaceable, at all times patient and kind, especially in difficult times. Thanks to exist and remain always by my side.

I would like to thank Matteo and Chiara, starting with the little things, for each message, for each weekly appointment, for every minute you have listened to me, to continue to be two irreplaceable friends, even during periods of absence.

I would like to thank those who are no longer in my life, because I like to think that your teachings have contributed to this work too.

ABSTRACT

The work involved the analysis and in-depth study of an auditory visual model to explain the formation of the taxonomic response and the fast-mapping abilities of children in early childhood. We started from an existing model that implements the visual and auditory categorization through the use of two self-organizing maps (SOM) and the association capacity, needed to learn the words, through Hebbian associative learning: the model is severely limited in the number of words that is able to learn without previously accomplishing a strong categorization process. Our proposal adds two major new features: first, the presence of a form of non-conventional, incremental training, in which the categories are presented in stages; second, the ability of the two self-organizing maps (visual and auditory) to grow during training phase (Growing-SOM) when they are no longer able to consistently represent categories. Some preliminary tests lead us to believe that the model is able to automatically adapt to the training set which is subjected, leading to strong performance advantages in the presence of a form of incremental training, most likely the type of "training" which is subject a child in the first twenty-four months of life. Finally, we conducted an in-depth study on the theoretical meaning of "errors of over-extension" during production tests: this made it possible to identify more carefully the nature of this type of errors and isolate more effectively the reasons why they take place within the model (categorization errors or associations errors).

CONTENTS

LIST OF FIGURES

LIST OF TABLES

1

INTRODUCTION

To understand the Turing model of 'the brain', it was crucial to see that it regarded physics and chemistry, including all the arguments about quantum mechanics..., as essentially irrelevant. In his view, the physics and chemistry were relevant only in as much as they sustained the medium for the embodiment of discrete 'states', 'reading' and 'writing'. Only the logical pattern of these 'states' would really matter. The claim was that whatever a brain did, it did by virtue of its structure as a logical system, and not because it was inside a person's head, or because it was a spongy tissue made up of a particular kind of biological cell formation. And if this were so, then its logical structure could just as well be represented in some other medium, embodied by some other physical machinery. It was a materialist view of mind, but one that did not confuse logical patterns and relations with physical substances and things, as so often people did.

— Alan Hodges, preface of The Computer and the Mind: An introduction to Cognitive Science by Philip Johnson-Laird.

The foreword by Alan Hodges reflects the simulation approach to which computer science is moving: if in the beginning the computers were seen as simple machines to perform functions and deliver quick results, with technological change, and the advent of increasingly complex models, many algorithms have been developed with the idea of solving problems in the same way they are solved by humans. This new point of view to pursue the same results has led to the emergence of new approaches in Artificial Intelligence (later called strong, precisely because of their simulative nature), which find their highest expression in Artifi-

cial Neural Networks. These models, from a strictly functional point of view, are part of the set of machine learning techniques together with many other classical optimization algorithms that are very distant from the functioning of the human brain. Moreover, the logical pattern of the latter, mentioned by Hodges, is far from being completely uncovered. We call psychology the kind of scientific investigation that studies the psychological, mental processes, and cognitive components in their conscious and unconscious.

A special branch of psychology is called experimental psychology and uses the experimental method in the investigation of cognitive processes. In recent years, a strong use of artificial neural networks has been done in experimental psychology: since it is difficult to identify analytically *how* cognitive processes are realized in the brain, the experimental simulation approach, based on scientific evidence, tries to emulate the operation of the first by obtaining the same type of results from models that are based on computational units similar to biological neurons. Through benchmarking and comparative techniques we can get valid indications on the development of complex cognitive processes.

The work done in this thesis has as main objective the development of a model that explains the incurring of a particular linguistic constraint needed to language learning in early childhood: the taxonomic constraint. This consists in the part of children's ability to learn, after a single events of association, that a given word refers to an object and to all objects of the same kinds (belonging to the same category). To investigate the origin of this type of constraint, and its implication in development of language in early childhood, we have been inspired by a model described in the literature by Mayor & Plunkett (2010), whose plausibility is undermined by some limitations, including the *inability* to learn new words after a previous categorization process.

With the aim to overcome this limitation, the major result achieved was the introduction of a new neural model for the simulation of visual and auditory cortex, able to evolve as a result of the introduction of new

categories during the training phase. The introduction of Growing SOM (replacing classical static SOM) at the base of the model was conducted starting from a technique illustrated in the literature in the mid '90: the method of Bernd Fritzke has been modified to obtain a type of expansion necessary to exceed the limitation of the model. The results obtained are very encouraging, bringing the model to have similar performance to the same architecture created with static larger SOMs, with necessarily fixed and previously established dimension. The dynamic expansion increases the plausibility of the model, providing a much desired feature, and giving credit to the way in which the ability to respond in a taxonomic way manifests itself in young children.

In the background, the work done has allowed us to develop useful visualization tool for studying the behavior of self-organizing map by a double point of view, both as vector and topological-organized spaces; furthermore, we re-defined the meaning of over-extension (during production test), identifying more accurately the origin of this type of errors in the model, isolating them more efficiently from random errors.

The thesis is structured as follows: in chapter 2, is conducted in-depth analysis of the SOM model originally proposed by professor Teuvo Kohonen. The Kohonen maps are the basis of visual auditory neural model discussed in detail in chapter 3: in the latter, we present the meaning of a linguistic constraint, the learning by Hebbian association and we illustrate some pictures of tools made and used for experiments. In chapter 4, they are brought to light the shortcomings of the initial model, with emphasis on new words learning limitations. In the same chapter we present the learning technique with incremental introduction of categories, the introduction of Growing SOM, and two expansion modes: at last, the results recorded during trials, before and after the introduction of the improvements. In chapter 5, we discuss some potential future development of the model. A selection of specific trial results could be found in appendix.

2

SELF-ORGANIZING MAP

Even though the Perceptron was just a simple but severely limited binary classifier, it introduced a great innovation: the idea to simulate the basic computational unit of a complex biological system that exists in nature.

In this chapter, it will be illustrated the functionalities of self-organizing-maps as neural network model. SOMs are one of the key elements of the visual uditive model presented in chapter 4. In this chapter, the reader will find an introduction to neural network model in section 2.1, the formalization of SOM model in section 2.2, the explanation of SOM training algorithm in section 2.3 (inspired by [11]) and examples of use in categorization problem in section 2.4 (taken from [6]). Finally, the last session (section 2.5) of this chapter, as in all other chapters of the thesis, contains a brief summary of what was said before.

2.1 NEURAL NETWORKS AS COMPUTATIONAL TOOLS

2.1.1 *Brief history*

In recent years, machine learning and cognitive science began to share tools and techniques to achieve different purposes. One of the tools most worth of attention are the artificial neural networks (ANN). We can date the birth of artificial neural networks in 1958, with the introduction of Perceptron [33] by Frank Rosenblatt. It was the first algorithm created to reproduce the biological neuron. Conceptually, the easier perceptron that you might think of is made of a single neuron: when it's exposed

to a stimulus, it provides a binary response, just as would a biological neuron. The easiest way to implement this simple classifier is to establish a threshold function, insert it into the neuron, combine the values (eventually using different weights for each of them) that describe the stimulus in a single value, provide this value to the neuron and see what it returns in output. This model differs greatly from the neural network involving billions of neurons in a biological brain. Shortly after his birth, the researchers showed the world the problems of Perceptron: in fact, it was quickly proved that perceptrons could not be trained to recognize many classes of input patterns. To get a more powerful network, it was necessary to take advantage of multiple level of units and create a *multilayers* perceptron, with more intermediates neurons used to solve linearly separable[1] subproblems, whose outputs were combined together by the final level to provide a concrete response to original input problem. Even though the Perceptron was just a simple but severely limited binary classifier, it introduced a great innovation: the idea to simulate the basic computational unit of a complex biological system that exists in nature. However, the need to study and understand how real neurons are bound to each other in the biological brain goes back to previous work.

2.1.2 *Cell-assemblies*

The most important work by Donald Hebb is *The organization of behavior* [7] of 1949. In this book Hebb combined for the first time data from the physiology of the nervous system with those of studies on human behavior, to which he added the personal experience derived from observation of the primates. His theory on learning association was born as an attempt to reconcile some seemingly inexplicable findings with the knowledge of the time, such as the fact that the human perceptual system can recognize a stimulus, such as a circle, although he sees it from different angles and with a form not exactly circular [38]. To solve this problem, Hebb introduced what is still known as the Hebb rule, or Heb-

1 This condition describes the situation in which there exists a hyperplane able to separate, in the vector space of the inputs, those that require a positive output from those requiring a negative output.

bian learning: according to this rule, «*when an axon of cell A is near enough to excite cell B and repeatedly or persistently takes part in firing it, some growth process or metabolic change takes place in one or both cells such that A's efficiency, as one of the cells firing B, is increased*». The combination of neurons which could be grouped together as one processing unit, Hebb referred to as "cell-assemblies". He also hypothesized that their combination of connections made up the ever-changing algorithm which dictated the brain's response to stimuli. This hypothesis will be basically confirmed in the following decades: a key aspect of neural models is precisely the topology of the network (especially in SOM), so that it is reasonable to think that the learning algorithm of some of them resides in part in the way the computational units can communicate with each other.

The Hebb rule and Hebbian learning together form the second key element of the auditory visual model that will be presented in chapter 3 section 3.3.

2.1.3 *Kohonen Maps*

Although the exact explanation of how certain cognitive processes develop in real brain is still a question that has not been given precise answer, neural networks as models start from the idea to simulate the interaction between artificial neurons, just like is done in the brain. A neural network is therefore a set of neurons interconnected by synapses: some of them receive input signals that can propagate to other unit/output with different intensity (possibly zero), depending on the configuration inside the neuron.

Different neural network models have been developed after Rosenblatt's Perceptron and its variants. As we said, as a computational tools, Perceptron has been initially discarded because it can solve a rather narrow class of problems (those linearly separable). The debate on research in this area was reopened only in the early '80, with the publication of the studies of the American scientist John Hopfield, working on models of pattern recognition. In those years, the Finnish professor Teuvo Kohonen introduced to the scientific community the Kohonen Maps, today known as Self-Organizing Map or SOM. This model consists of a group of

neurons topologically arranged in a lattice, able to self-organize through unsupervised learning. The appearance of unsupervised learning is often sought after, especially in models that want to be psychologically (and neurobiologically) plausible. In essence, the learning of the ANN can occur in two modes: the supervised one makes use of the label of input patterns examined during training. Many models make use of this technique because they inherit, from the world of machine learning, the idea to minimize the accumulated error (calculated by checking systematically the result obtained by the algorithm of classification on the input processed during the training phase). The technique of unsupervised learning, used by algorithms such as the Kohonen Maps, is more independent during the training: that is to say, the training algorithm does not need a training set "labeled", as the model discovers automatically similarities, with the "downside" of not being able to choose in advance the number of classes of the problem in the output space, but having to define them on the basis of the output data from the model itself. One of the biggest criticisms of the supervised learning algorithm consists precisely in their need for a labeled training set: in fact, there is no evidence that cognitive processes that take place in the biological brain make use of error correction techniques. Finally, self-organizing maps are different from other artificial neural networks as they apply competitive learning. The goal of learning in SOM is to specialize different parts of the map's lattice to respond similarly to particular input patterns. This is partly motivated by how visual, auditory and other sensory information are handled in separate parts of the cerebral cortex in the human brain: in this sense, as we said before, the SOMs are a neurobiologically plausible model. SOM model proved to be able to function very well with input data with high dimensionality, as a technique of unsupervised learning capable to produce a discretized representation of the input space with reduced dimensionality.

In the next sections it is presented the model originally proposed by Kohonen, starting from the mathematical formalization, the learning algorithm, and the technique used to categorize new stimuli. Later in this work, it is made extensive use of the concepts introduced in the following sections: the model that is presented in those pages has its bases on two distinct SOM, a map that simulates the visual, the other auditory cortex

in real brain. For those who are familiar with the structure of the neural network topologically organized proposed by Kohonen, please refer to chapter 3.

2.2 FORMALIZATION

2.2.1 *Model description: a topologically organized space*

As we said, SOM consists of a set of neurons also called nodes or units. Each neuron is associated with two information: a weight vector of the same size of the input data and a position in the grid on which are placed the neurons. Because of this characteristic, a SOM consists of an n-dimensional vector space, with n well-defined dimension equals to the number of elements (features) of the input vectors, filled with m point (neurons): the space is also topologically organized, with topological order defined by the location of each neuron on the map's grid. The grid can take many forms, usually linear form such as rectangular or hexagonal. In Figure 1 are shown some possibile grids. For examples, given a rect-

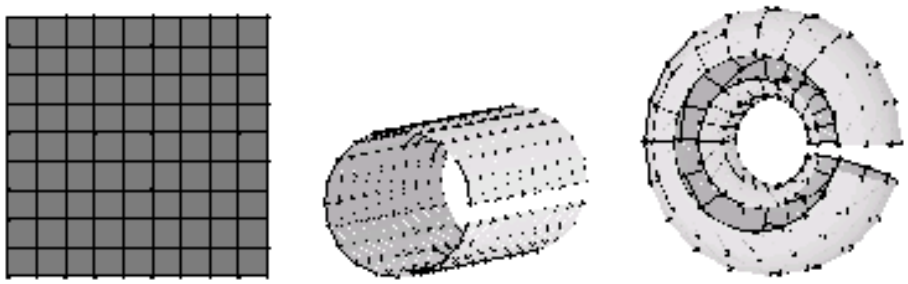

Figure 1: SOM possible grid

Each point represent a neuron (computational unit); lines are synapses between neurons and form with them what is called *lattice* of map.

angular lattice $m \cdot l$ with $m \cdot l$ neurons, neuron n is defined by its weight vector w_n and its coordinates x, y on the map. Given weight length k, input vectors have k elements describing the input with k different characteristics.

2.2.2 *Performance indicators of model*

The quality of the self-organizing map can be assessed by two major statistics:

- Quantization Error (aka Q. Error or QE): is computed by determining the average distance of input vectors to the prototype vector of the BMU that represents them. In formula,

$$QE = \frac{\sum_{i=1}^{i=N} \|x_i - W_{c_i}\|}{N} \tag{1}$$

- Topological Error (aka T. Error or TE): is the most simple measure for topology preservation. For all input vectors the respective BMU and the second-BMU are determined. If these two are not adjacent on the map lattice then this is considered an error. It is calculated as

$$TE = \frac{\#errors}{N} \tag{2}$$

  where 0 means perfect topology preservation.

These indicators will be useful when you will be presented with two SOMs at the base of the visual auditory model presented further. Especially the QE play a fundamental role in Hebbian learning. For more details, refer to section 3.3, chapter 3.

2.3 TRAINING

SOMs apply competitive learning: the idea is to strengthen the weight vectors of the neurons that respond more strongly to a particular pattern (also called best-matching unit or, shortly, BMU), while systematically increasing the likelihood that the same neuron, in subsequent iterations, will respond with greater "emphasis" to the same input pattern or a slight variation. To obtain this result, an activation and update rule are defined: the first allows us to decide which neuron is activated with greater "emphasis", the second allows us to reinforce the weights of that neuron[2]. After the presentation of all pattern in training set, a training-epoch ended.

2 Also its neighbors: more details later in text.

The weight vectors of neurons are initially randomly set[3]: furthermore, given a fixed epoch e, the update of the weight vector in e of each neuron can occur in two distinct moments, depending on whether you make use of the classical learning algorithm or batch variant. We will return on the learning algorithm later in Equation 2.3 and subsection 2.3.1.

After n epochs in which all input patterns are randomly selected from the same set of training, neurons on the map are organized in such a way that similar areas of the map correspond to similar patterns of the input space. Then weight vectors of the map are frozen: to categorize, a new stimulus is presented to the map and, using the same rule employed in training to determine what was the Best-Matching Unit, we can establish what is the BMU that is activated with greater intensity: depending on the position of the neuron on the grid, it is possible to read on map the category which the stimulus belongs to. The capacity of the map of categorizing depends on multiple factors, such as the number of training epochs. Although the training operation appears to be simple, there are many parameters involved in successful learning and which determine the quality of the trained map: unfortunately, there are no precise rules to better adjust these parameters. In the next paragraph, we will discuss about the most important parameters and rules should be laid down for training.

As we said, the results of the training phase are closely related to the number of training epochs. However, there are at least two other parameters that affect in a substantial way how the neurons of the map are self-organizing: these are the *neighbor function* and the *learning rate*. The first defines how the proximity between two neurons affects the updating of the weights. More precisely, we have said that the update of the weights involves substantially BMU of the input pattern, but also its neighbors. The neighbor is a function that defines how the proximity to the BMU affect updating: it could be a constant function (fixed value) or vary over time narrowing more and more the "radius" of neurons involved in the

3 However, the weights of the neurons could be initialized either to small random values or sampled evenly from the subspace spanned by the two largest principal component eigenvectors.

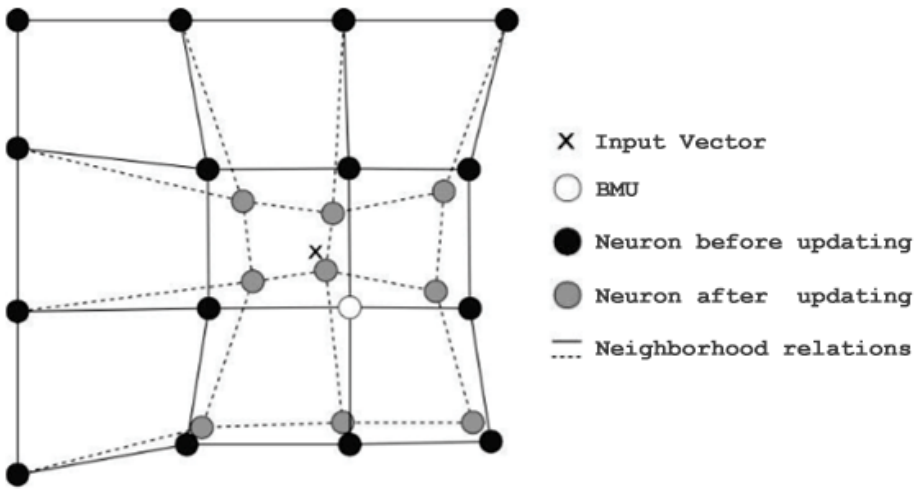

Figure 2: Example of SOM weights updates

A demonstration of weights update result respect to original unit position.

update[4]. weight vectorThe second parameter is called the learning rate and is a monotonically decreasing learning coefficient: it is commonly called α.

FORMULAS All we said in the previous paragraph could be written in two different formulas. The first is the activation rule. Given the Euclidean distance function from $x(t)$ and $W_k(t)$ (where t is the epoch number, x the input pattern, $W_k(t)$ the weight vector of neuron k in t) as

$$d_k(t) = ||x(t) - W_k(t)|| \tag{3}$$

we define BMU the neuron b such that:

$$d_b(t) = \min_k d_k(t); \tag{4}$$

when the d function is defined as said above. In other words, each time that an input pattern is presented to the map, the activation rule says that the BMU is the neuron such that the Euclidean distance between its weight vector and the input pattern is minimal.

4 This is not to say that the update does not involve in every epochs all the neurons of the map. The neighbor affects the delta that is added / subtracted to each component of the weight vector of each neuron. For neurons sufficiently distant update turns out to be negligible, but conceptually the update involves in every epochs all the neurons of the map.

The second formula is the update rule, used during training to update weight vectors of neuron. This rule is activated for all k neurons of the map, with the neuron b that plays the role of BMU, chosen by the rule defined above. During every epoch training, every time[5] a stimulus is presented to the map, the neuron c is chosen as BMU using activation rule. Subsequently, the following operation is performed.

$$W_k(t+1) = W_k(t) + \alpha(t) \cdot h_{ck}(t) \cdot [x(t) - W_k(t)] \tag{5}$$

where:

- $t+1$ is the current training epoch number (t is the previous);
- W_k is the weight vector of neuron k;
- $\alpha(t)$ is the learning rate value at epoch t;
- $x(t) - W_k(t)$ is the difference between weight vector of neuron k and input x in epoch t;
- h_{ck} is the function that uses the value of the neighbor in epoch t to modulate the updating: the results of this function are calculated based on the distance between the BMU and the neighboring neuron k in the map's grid. Euclidean distance is commonly used. The h function could be defined as:

  $$h_{ck} = e^{-\frac{||cord_k - cord_c||^2}{\sigma(t)^2}}$$

  in Gaussian form;

  $$h_{ck} = \begin{cases} 1 & ||cord_k - cord_c||^2 < \sigma(t) \\ 0 & otherwise \end{cases}$$

  in piecewise form;

  with $cord_k$ and $cord_c$ coordinates between neuron k and BMU c and $\sigma(t)$ neighbor function value in epoch t;

weight vectors

5 Or one time only at the end, depending on which algorithm is used.

There are at least two different versions of the training algorithm: in the first the weights are updated after every pattern presentation. In batch mode, update are computed all in one time, after all pattern was presented. In next page is shown the pseudo-code of the classical version (Algorithm 1).

Algorithm 1: SOM classical training algorithm.

Data: Training set
Result: Trained map
Initialize the weight vectors, $W_i \in R^n$;
$t \leftarrow 0$;
for epoch = 1 *to* N_{epochs} **do**
  for input = 1 *to* N_{inputs} **do**
    $t \leftarrow t + 1$;
    for k = 1 *to* $K_{neurons}$ **do**
      Compute distance d_k using Equation 3;
    end
    Compute BMU for current input using Equation 4;
    for k = 1 *to* $K_{neurons}$ **do**
      Update weight vectors W_k using Equation 5;
    end
  end
end
return

2.3.1 *Batch training algorithm*

In the batch variant of the SOM algorithm the updates are deferred to the end of a learning epoch, i.e., the presentation of whole training set and the new weights are computed using these:

Batch update rule

$$W_k(t_f) = \frac{\sum_{t'=t_0}^{t'=t_f} \tilde{h}_{ck}(t') \cdot x(t')}{\sum_{t'=t_0}^{t'=t_f} \tilde{h}_{ck}(t')} \tag{6}$$

Batch Euclidean distance

$$\tilde{d}_k(t) = \|x(t) - W_k(t_0)\| \tag{7}$$

Batch activation rule

$$d_c(t) = \min_k \tilde{d}_k(t); \tag{8}$$

where t_0 and t_f stand for, respectively, the beginning and end of the current epoch. The learning rate factor is explicitly present in the batch update equation and does not need to be parametrized. The rule involves all neurons, but depending on neighbor function h, a neuron could not be affected during an epoch: in this case, $W_k(t+1) = W_k(t)$. The batch variant can only be applied when the whole set of input data is present. Finally, since the weights are not updated immediately there is no dependency on the order of the input vectors. In next page is shown the pseudo-code of the classical version (Algorithm 2).

Before proceeding with some examples of use of the SOM in real problems it is necessary to highlight another thing about the learning algorithm (regardless of the version used), linked in some way to the learning rate and neighbor function: starting from a state of complete disorder, the SOM algorithm gradually achieves an organized representation of the input space, provided that the parameters are chosen properly. This adaption process should be decomposed into two phases: an ordering phase followed by a convergence phase.

Ordering phase It is during this first phase of the adaptation process that the topological ordering of the weight vectors takes place. This ordering phase is relatively short in comparison to the second phase. Large values for the neighborhood radius and learning rate should be used, such that the neuron's weights initially take large steps all together toward the area of input space where input vectors are occurring. These values then should decrease to their tuning values and, consequently, the neighborhood decreases to encompass only the closest neighbors.

Convergence phase This phase lasts for the rest of the training or adaptation process and is necessary to fine tune the network and there-

Algorithm 2: SOM batch training algorithm.

Data: Complete training set
Result: Trained map
Initialize the weight vectors, $W_i \in R^n$;
$t \leftarrow 0$;
for epoch = 1 *to* N_{epochs} **do**
  Interpolate new value for $\sigma(t)$;
  Reset numerator and denominator of Equation 6 ;
  for input = 1 *to* N_{inputs} **do**
    $t \leftarrow t + 1$;
    for k = 1 *to* $K_{neurons}$ **do**
      Compute distance d_k using Equation 7 ;
    end
    Compute BMU for current input using Equation 8;
    for k = 1 *to* $K_{neurons}$ **do**
      Accumulate numerator and denominator of Equation 6;
    end
  end
  for k = 1 *to* $K_{neurons}$ **do**
    Accumulate numerator and denominator of Equation 6;
  end
end
return

fore provide an accurate statistical quantification of the input space. During this phase the weight vectors converge to their *correct* values. For this, the neighborhood should be fairly small, encompassing only the immediate neighbors. This also applies to the learning rate, such that the magnitude of the weight updates is very small. The convergence phase is usually several times longer than ordering phase.

2.4 EXAMPLE OF USE IN CATEGORIZATION PROBLEM

2.4.1 *Color categorization*

SOMs are capable of grouping by similarity high dimensionality data and display groups in a reduced number of dimensions (i.e. two dimensions). Suppose we have available the color scale shown in Figure 3. We divide this color scale in 25 parts, each colored with a different color of the scale. The various parts are our input patterns. The task is to group them by color, in a way such that similar parts are mapped by near neurons on the map (we can think about different brightness as different categories). Each of the input patterns is mathematically represented by a triple RGB, then we want to categorize 3-dimensional inputs.

Figure 3: HSV colormap scale from MATLAB

Suppose we have available a map with 100 neurons an hexagonal grid (10x10), such as that shown in Figure 4.

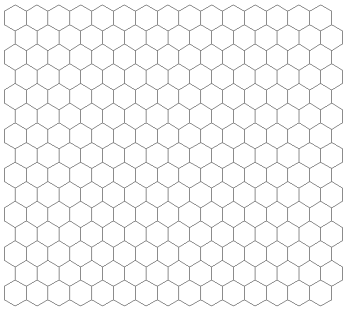

Figure 4: SOM structure

We initialize the weights of the map randomly, aligning them with averages calculated on samples of input patterns taken at random from the set of 100 defined above: we can see the resulting map in the left upper corner of Figure 5. The various units were filled with RGB colors that correspond to the respective weight vectors.

In the same figure we can see resulting map during training at 5, 10, 15, 20, 25 epochs, respectively. In this trivial example of categorization, it is shown clearly how the weights of the map's units align with the input patterns. In addition, you can clearly note the distinction between the ordering phase and the convergence phase: in fact, the first half of the training brings the map to align themselves in a position that will maintain almost unchanged in the second half of the training, except for some slight modification (i.e. see angle at the bottom right of epoch 15-20-25).

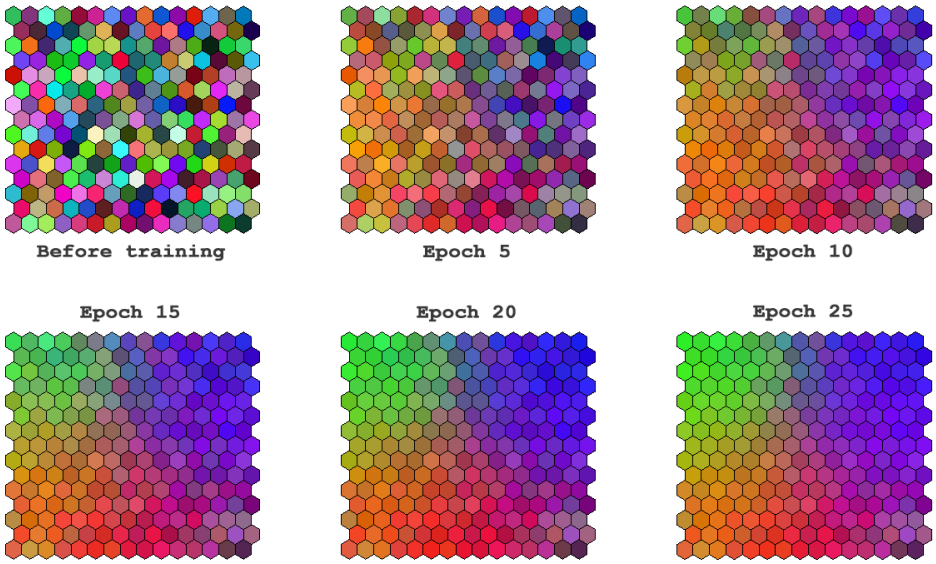

Figure 5: SOM epoch training

Images from left downer corner to right show weight vectors during map training. In left upper corner map training status reveal lots of noise: each unit is colored with using respective weights as RGB values representation. After 25 epochs, weights are aligned with input patterns: organization is reached.

2.4.2 *Visual pattern categorization*

Imagine to have a series of images, not necessarily complex: for simplicity, suppose we have available only the outlines of well-defined shapes, black on white. There are many image processing techniques that can encode the edges of a figure within a image: although the input encoding problem is fundamental to the study of neural networks, we assume that

the contours are defined by pixels. The information that describes these pixels, since they are all blacks, is linked only to the position in the grid: setting an origin, we can imagine our figure as a polygon traced by a continuous line (in reality, segmented to arbitrarily thin grain) in a Cartesian plane in which every point is defined by (x, y) coordinates. In Figure 6 we see some examples. These coordinates, set a way by which to order them, they can be concatenated into a vector, which becomes the coding of the image, shown in the figure below each view through coordinates. In this scenario, it was decided to encode blacks points, and reading them by column from left to right, the x, y coordinates of each point are concatenated consecutively. Alternatively, you can encode the white spots, or make further changes to make the coding insensitive to scaling transformations, rotations, etc. Below, you can see a classification of visual patterns, encoded using three different values. The SOM, in this example can be a useful categorization tool and pattern recognition, as similar areas will be able to categorize visually similar animals.

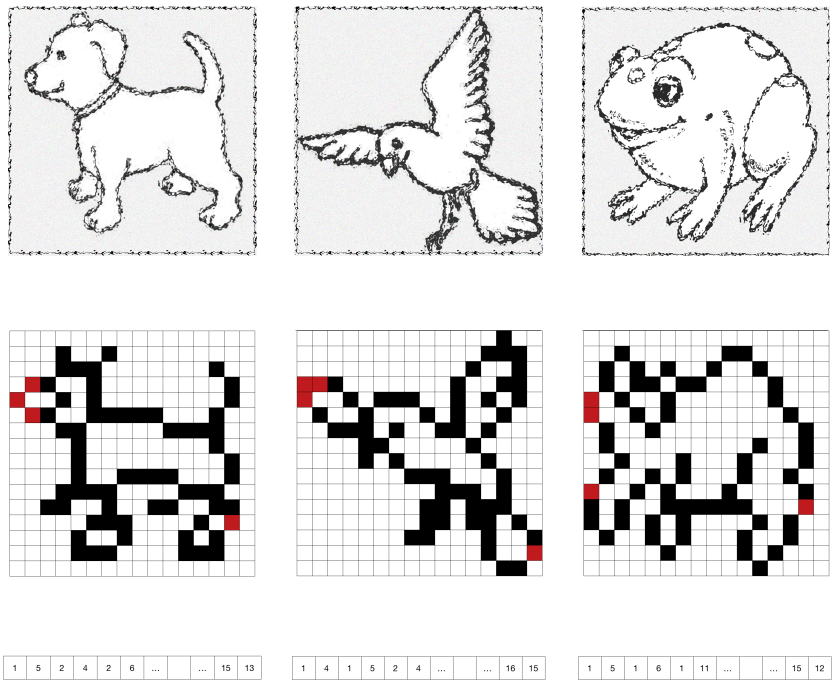

Figure 6: Visual pattern encoded for SOM training.

In Figure 7, we see the evolution of a SIM that categorizes visual pattern defined along three dimensions: these correspond to the size, in cm, on

the x, y and z of the animal's profile seen from the front. The map is displayed every 10 training epoch, for a total of 100 training epochs with standard parameters defined by the library SOM Toolbox. The categories defined in the epoch 80, 90 and 100 on the map are displayed in the screenshot larger than the earlier epochs. The colors, as far as possible, are aligned with the diversity between categories: we have confirmation of the categorization ability of the map also thanks to the grouping of cold and warm colors in different areas, in addition to the names of the categories that defined by each neuron.

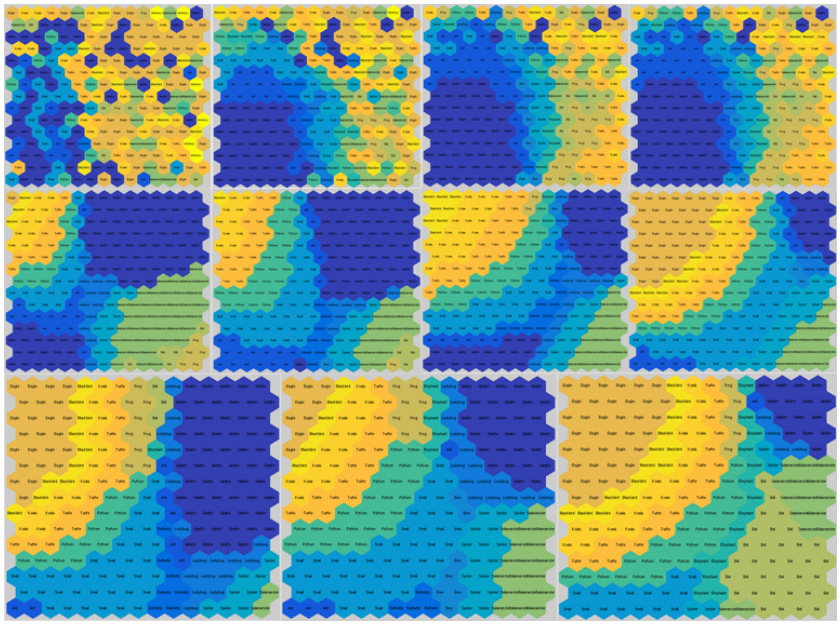

Figure 7: SOM evolution during standard training with category boundaries highlighted by different colors.

2.5 SUMMARY

In this chapter we introduce some key concepts necessary for visual auditory model described in chapter 3, starting from some historical notes related to the birth of the interest and the study of ANN in section 2.1 to the formalization of the SOM in section 2.2. In section 2.3 we have provided a detailed description of the algorithm of learning (the classic version and the batch) and, finally, in section 2.4 we presented a simple example of using the color classification somtoolbox and MATLAB. In the

next chapter, we will present the visual auditory model created to explain language acquisition in young children, showing how this is based on the proper functioning of the SOM.

3

VISUAL AUDITORY NEURAL MODEL

Contents: ***tools*** *to display* ***Voronoi tessellation*** *on the lattice of a SOM, with a table with relative distances between the defined cateweight vectorgories, useful for viewing the dual nature of the space defined by the SOM and analyze the topological organization of the categories corresponding to the distance between the prototypical pattern from which they are defined; formalization of a* ***testing*** *algorithm in* ***production*** *that identifies accurately the* ***causes of over-extension errors****, with detailed analysis of the origin of the error.*

In this chapter, we present the neuro-computational model using SOMs that accounts for the emergence of taxonomic responding and fast mapping in early word learning, as well as a rapid increase in the rate of acquisition of words observed in late infancy. This model was developed by Julien Mayor & Kim Plunkett ([21]): as research work, we retraced their steps rebuilding the model from the beginning, to test the limits and try to overcome them. In section 3.1 we retrace the reasoning by the original authors leading to the formalization of the model, talking more specifically about the problem and linguistic constraint that guide word-learning in early childhood; section 3.2 describes the set up of the two self-organizing maps simulating the visual and auditory cortex; in section 3.3, it's introduced Hebb rule and the mechanism of learning by association; in section 3.4 it's presented evaluation factor and production-testing algorithm created to explore precisely the causes of the errors committed by the model; section 3.5 shows some demos of the model in action, with

pictures and results related with discussion about limitations. Finally, section 3.6 contains a brief summary of what was said in the chapter.

3.1 EARLY WORD LEARNING

3.1.1 *The problem*

A central question in early lexical development is how infants learn to understand the meaning of words. In a typical labelling situation, the caregiver points at an object (i.e. a dog) and says "Look, this is a dog!". Then magic happens: he is able to understand that the word "dog" refers not to the size, the color, in that particular dog or to a part thereof, but to *dog and all the other dogs* (Plunkett, [24]). Moreover, dog could also refer to "the dog and his bone", or "Mommy petting the dog" or "the dog under a tree". In other words, objects are often found in spatial, causal, temporal or other relations with other objects, so what prevents the child from thinking that the label refers to the objects that are related? [17].

The research for the past several years has come to the conclusion that language learners make use of special linguistic constraints: these play a key role in early words learning, allowing the children to learn more easily the associations between words and objects, thus explaining the growth of vocabulary and the consequent ability of learning new words in adulthood. We may think that «contraints guides children's initial hypotheses and eliminates numerous hypotheses from consideration» (Ellen M. Markman, [17]).

The most important constraints are Whole Object Constraint (for short WOC), Taxonomic Constraint (for short TC) and Mutual Exclusivity (for short ME). In the next few paragraphs we explain the meanings of these constraints and how they provide support lexical learning in early childhood. According to Plunkett [24], these three constraints are not innate: as opposed, can emerge from processes of associative learning. We find confirmation of this statement in the model that will be presented in the following pages (initially introduced in [21]). In section 3.5 we'll talk profusely about limits of the model, testing its solidity and analizing results.

In chapter 4 are shown extensions and additions that led to overcome some limitations. Before formal explanation of the model (section 3.3), we analyze briefly the meaning and the contribution that the three linguistic constraints above lead to new words learning in early-childhood.

3.1.2 *Linguistic constraints*

WHOLE OBJECT CONSTRAINT WOC occurs when children associate a label to an object without making the mistake of labelling a specific part of it. In other words, labels are always associated with the whole object. In [21], Mayor & Plunkett argued that the abstract coding scheme adopted for visual objects (Posner & Co, [28, 29, 31, 30]), allows them to remain agnostic as to the types of attributes involved in category formation, so that input categories can be interpreted as instantiating *family resemblance* characteristics defined across, say, either functional or perceptual attributes: however this involves accepting more limitations on visual recognition. Categorization of visual objects remains one of the big challenges despite an enormous progress in this area. The principal problem is the selection of the features that can represent a visual object in a way invariant to factors like rotation, scaling, changes in illumination and the viewpoint [26]. Also in speech recognition, a sophisticated phonological knowledge must be taken in account: in [37], White & Morgan demonstrated that toddlers as young as 19 months have highly detailed and apparently adult-like lexical representations. We will return later in this chapter on these consideration, talking about visual-auditory capabilities and consequences on recognition performance.

Returning to WOC, researchers agree in thinking that neural networks in the hippocampus stores episodic memories and that their computational / architectural structure resembles that of an auto-associator [35]. An auto-associator is capable to compute correlations in the activity of different components of the input signal, and adjust the connections appropriately (Plunkett, [24]). We have already mentioned the auto-associator, without providing a specific name. In the introduction to chapter 2, we briefly mentioned Hopfield works on mechanisms of pattern recognition:

Hopfield introduced his idea of recurrent artificial neural network[1] in 1982 [8]. His ANN serve as content-addressable memory systems with binary threshold nodes, providing a model to understand human memory. A simple associative implementation of the WOC might exploit an auto-associator: the WOC can then be construed as encoding a pattern of correlations between a package of linguistic features and a package of visual features in a compound audio-visual stimulus. Despite the WOC is topic of interest, the constraint that emerges from the analyzed model in chapter 3 and chapter 4 is second on the list, called TC or Taxonomic Constraint. However, for those interested WOC auto-encoder is presented by Plunkett in [24].

TAXONOMIC CONSTRAINT They are two definitions of the taxonomic constraint: at first glance may seem similar but stronger definition brings an extra feature to be satisfied. The weak definition assumes that

> «*labels refer to objects of like kind rather than objects that are thematically related*» - Markman, [18]

In other words, labels refer to objects that belong to the same taxonomic category, where a category can be defined in terms of visual features (visible or hidden, such as "color" or "tail size") or functional relations (dynamic or abstract, such as "belong to the animal world"). The strong definition says that

> «*when infants embark upon the process of lexical acquisition, they are initially biased to interpret a word applied to an object as referring to that object and to other members of its kind*» - Waxman, [36]

In this form, TC is secretly linked to the ability to *quickly* obtain the same general and effective connection expressed by weak definition of TC, between the name of a category (i.e. *dog*) and "objects" that are

1 Also known as R(A)NN, is a class of artificial neural network where connections between units form a directed cycle. This creates an internal state of the network which allows it to exhibit dynamic temporal behavior. The Hopfield network is of historic interest although it is not a general RNN, as it is not designed to process sequences of patterns. Instead it requires stationary inputs. It is a RNN in which all connections are symmetric.

visually recognized as part of the same category (i.e. specific, possibly different, dogs). Practically, children that use the linguistic taxonomic constraint defined strong, infer that every object that belongs to the same category is called by the same name *after a single labelling event*. This ability is also known as *fast-mapping*. We will return soon on this aspect.

The studies in [18] of Markman & Hutchinson tested both the taxonomic (following the weak definition) and whole object assumption. To test this, they conducted a series of studies each of which compared how children would organize objects when they were not provided with an object label versus when the objects were given a novel label. From the results of their experiments it follows that even though children consider thematic relations good ways of organizing objects themselves, they do not consider thematic relations as possible meanings for words. Thus, when children believe that they are learning a new word, they shift their attention from thematic to categorical organization.

To implement the taxonomic constraint, you must define a model that creates a unique association between a label referring to a category and more individuals of the same category. The first model introduced by Plunkett, Sinha, Møller & Strandsby in [27], is an auto-encoder learner, which has computational properties similar to that of the auto-associator. We will not go into detail in the explanation of this model: it is enough to know that it is not able to meet the definition of strong taxonomic response. In fact, it requires exposure to multiple object-label tokens, without presenting the ability to fast-mapping required to justify the formation of the taxonomic response[2]. In order for this kind of one-shot learning and generalization to occur, Plunkett & Co concluded that the learner must have a sort of *prior knowledge* of the category boundaries.

In 2010, Mayor & Plunkett developed a new model based on SOMs [21]: this model provides a strong taxonomic response. We will profusely talk about its formalization in section 3.2. Let's first briefly present the third constraint.

2 The strongest form of taxonomic constraint, according to tests conducted on the ability of categorization of children, provides the ability to quickly get the taxonomic response, possibly even after a single labeling event.

MUTUAL EXCLUSIVITY Imagine that a child sees two objects in a scene, like a dog and a bottle, and know the name of the word "dog": if he hears a new word (i.e. "bottle"), he automatically associates it with the new object in visual scene, although it is known in adulthood that each object generally has at least two names[3] which refer to it. Conceptually, the constraint that helps him is the mutual exclusion: in [19], Markman & Co have shown that 15- to 17-month-old infants, upon hearing a novel word, will search for an alternative object if the only object they can see is name-known. So the child assumed that an object can only be associated with a single name. The mechanism that drives ME or the age when ME is first used remains a matter of dispute. In the next section, we will explain in detail the configuration of the visual-auditory model by Mayor & Plunkett.

3.2 VISUAL AND AUDITORY SOMS

The auditory visual model that we have analyzed and implemented makes use, as already mentioned, of two distinct SOMs. Both maps have a hexagonal grid[4] composed of 625 neurons, as shown in Figure 8. As discussed in chapter 2, each neuron has an associated weight vector: its size is tied to the input encoding, consequently is different for the two maps. Before speaking of the two maps and how they work, we present the encoding of visual and auditory input patterns.

3.2.1 *Visual input patterns*

Objects are represented as distorted dot patterns. It was difficult to retrace the steps of Mayor & Plunkett in creating patterns as there are many ways to generate random visual patterns. To stay faithful to the standards of diversity needed to justify maps behavior during experiment, we made use of a modified version of the algorithm used by Posner and colleagues in [31], which uses the Gaussian functions to distort patterns. As usual,

3 We're talking about common nouns but also proper noun: it's difficult find an object in word to which you can only relate with one name.

4 Hexagonal grid means that each neuron has 6 topologically closest to distance 1.

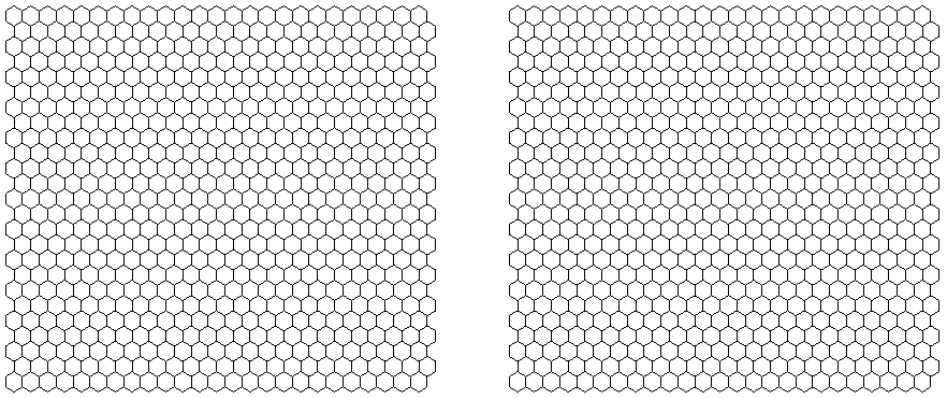

Figure 8: Visual and auditory SOM in word-learning model.

generating random mathematical objects is always a more difficult task than you can imagine. The original article by Posner and collegues (a really hard work that starts in 1962, see [28, 29, 31, 30]) proposes distorted nonsense dot patterns by means of three different types of statistical rules. The level of distortion was calculated from the number and probability of the cells to a given dot which could move and was expressed in term of uncertainty. However, to simplify and according to what was achieved by Mayor & Plunkett, we created 100 patterns (prototypes) by randomly distributing 9 dots over a $30 \cdot 30$ square. We then generated 24 tokens in each category, consisting of 8 tokens at each of 3 levels of distortions. Distortions are achieved by moving each dot by an amount drawn from a normal distribution with 3 different standard deviations (i.e., 10, 11, 12[5]) respectively. Prototypes were not included in the training set.

IN DETAIL Precisely, the algorithm used generates a vector of 18 not repeated integer values between 1 and 30: the indexed values in even positions are the x coordinates of the points to black in $30 \cdot 30$ square, the

5 These values are different from those used in [21]: however, the model has so many parameters for which it was necessary to perform a very high number of tests before it reaches an plausible equilibrium situation also useful to assess performance. The distortion patterns (exemplars) used in training / testing is a fundamental aspect of this tuning phase. According to analyzes carried out after the assembly of the model, parameters could conceivably vary with other and lead to the same conclusions.

odd values the y coordinates[6]. Moreover, the points are generated in such a way not to be overlapped (i.e. the 9 pairs generated are pairwise disjoint, equivalently have at least one coordinate different). One of the problems you have to deal with when you have to generate arbitrarily complex objects distributed as uniformly as possible in a small space, is to study the relative maximum distance that can exist between these: this value is hard to find, and it is also difficult to find a way to exploit it to generate patterns sufficiently different[7] from each others. If the portion of the space is not very wide and with many dimensions, generate sufficiently different objects can become an expensive operation to be carried out randomly: in fact, the generation can proceed for an arbitrarily long time (enough to have memory to hold computation partial results), however, the evaluation of patterns more distant between each others can become complex to manage for a sufficiently high number of random patterns generated. The problem can be reduced if we do not just consider as visual features the keypoints of the figures but also features related to colors or nuances[8], according to what is done in problems of visual analysis. To simplify, as the authors did in the original model, the implementation is limited to assessing as visual features only the keypoints of the image.

PROTOTYPES GENERATION The algorithm created for the visual prototypes generation takes as input parameter a threshold value (optionally expressed as a percentage, in our implementation is set to 0.45): this threshold is used to calculate the minimum distance *required*[9] between

6 To remain faithful to the one-dimensional representation of the input, we have adopted the following formalism: each visual pattern (prototype and example) is represented by a vector of 18 elements where the elements correspond to the 9 coordinates blackened points in the grid, in order, $[x1, y1, x2, y2, ...x9, y9]$ etc.

7 In all this work, unless otherwise specified, we associate the concept of diversity to the concept of Euclidean distance because the Euclidean distance is the only unit of measurement understood by self-organizing maps.

8 The increase in the number of features is reflected in the increase in size of the space, ergo an equal prototypes number is easier to distribute in a uniform manner, in a space with several dimensions.

9 Such that we can accept the generated prototype since it is sufficiently different from all others.

the prototypes generated. To fulfill any measure of comparison, the maximum distance between two prototypes must be taken into account. This can be calculated in different ways: we have taken as an approximation of this value the distance between the two patterns shown in Figure 9 starting from the left. The algorithm, in our example, estimates 45% of the maximum distance defined and uses this value as a threshold. For each generation of a new prototype, it compares the distance of this with each of those generated previously: if the distance is greater than the threshold calculated earlier, then it proceeds with two consecutive operations of shuffling of the coordinates, which attempt to move the last generated prototype away from the nearest prototype for a fixed predetermined times, after which it passes to the generation of the next prototype. The whole process is iterated for $10 \cdot n$ with n the number of prototypes required. After the generation, n prototypes maximally distant are selected and returned. We have also created various tools to display maps evolution and prototypes generated (with their distortions): the latter makes use of the function `scatter` from MATLAB standard library to draw a closed line between the points and the function `patch` (always from MATLAB) for coloring the defined area. The result of these operations is shown in Figure 9 (from the third to the last image) In Figure 10 is shown instead a display screen 25 prototypes and Figure 11 the display of a prototype and the respective distortions.

A wrong choice The problem of encoding the visual patterns was initially addressed using a direct binary representation form: that is to say, each visual pattern identified as binary vector (i.e. 900 cells), with black point set to 1. This road is definitively the wrong one: indeed introduces manifold problems, first of all to define a distance function that reflects the diversity between two patterns. Those which are sufficiently distant result too similar if we evaluate the distance because the information contained in binary vectors is too weak (many values set to 0 are confused as similarities, and also numerically the position of 1 does not affect the calculation of distances for many available distance functions.). To avoid running into these problems you need to remember the nature of features.

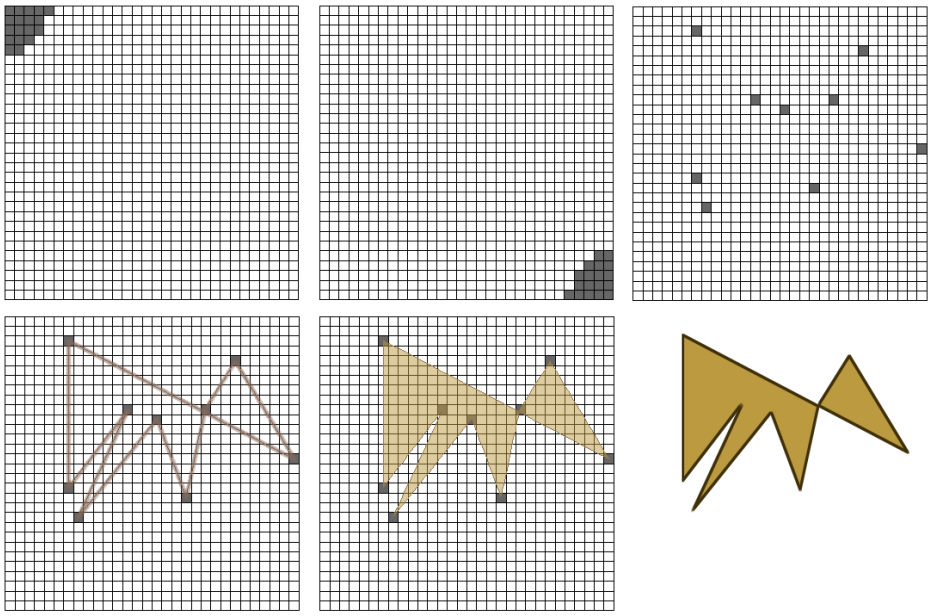

Figure 9: Twenty-five visual input patterns used as prototypes

In the two left-upper to right figure we see pattern used for maximum distance. In the other four ones we see creation of a pattern, from dot position to colored figure.

3.2.2 *Auditory input patterns*

For the auditory part, the authors of the model made use of advanced sound analysis to obtain pattern representations consistent with what is known about phonetic skills of young children. Omitting details about the technique exploited by Mayor & Plunkett[10], we just rebuild auditory patterns using the same number of dimensions and random method equals to the one used for visual pattern, with the objective to have a placeholder for the role of "spoken words" to use with the auditory map.

10 In [21], they describes a method to collect recorded spoken words by women (potentially to evoke the maternal voice that the child should heard more often), with introduction of high and low variance distortions following a classical $80-20$ law to mimic possible pronunciations of other people. Following, these sounds were combined, discretized and normalized so as to obtain likely complete (that maintained much information as possible) representation of heard words, as intensity of frequency calculated by exploiting the Mel scale.

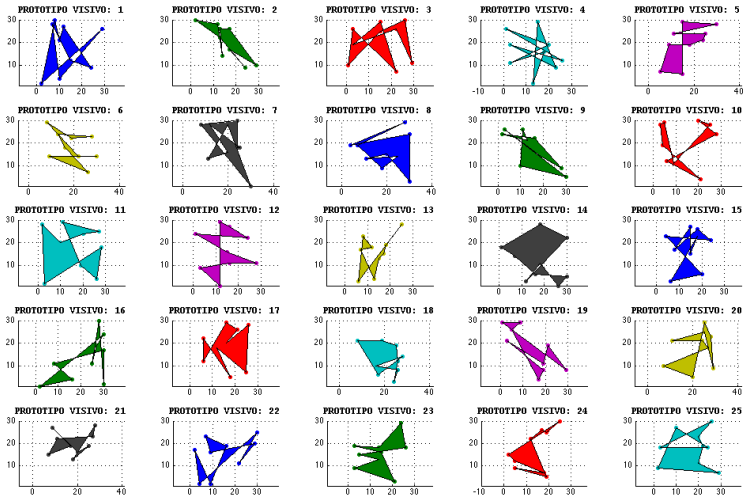

Figure 10: Twenty-five visual patterns used as prototypes

From each of them have been generated 24 distortions, using different Gaussian sigma values: the pattern are shown in 30x30 grid using `scatter` and `patch` functions by MATLAB. At first glance they seem very different from each other, for more details see the caption.

In particular, auditory patterns are composed of 28 values ranging between 1 and 10. In Figure 12, it is shown a sinusoidal representation which was created to distinguish them from visual patterns.

3.2.3 *Input offline testing*

After the creation of prototypes and related distortions, many tests have been conducted on the patterns generated in order to study sparsity and the conformation of the data. We did three main tests on generated input patterns:

- Test to see how close are the prototypes from each other. Among all, the statistics provided by this test generated prototypes include minimum average and maximum distance reached between each prototype, the upper bound of the distance reached (see above for more details) and the average distance normalized to the upper bound expressed as a percentage.

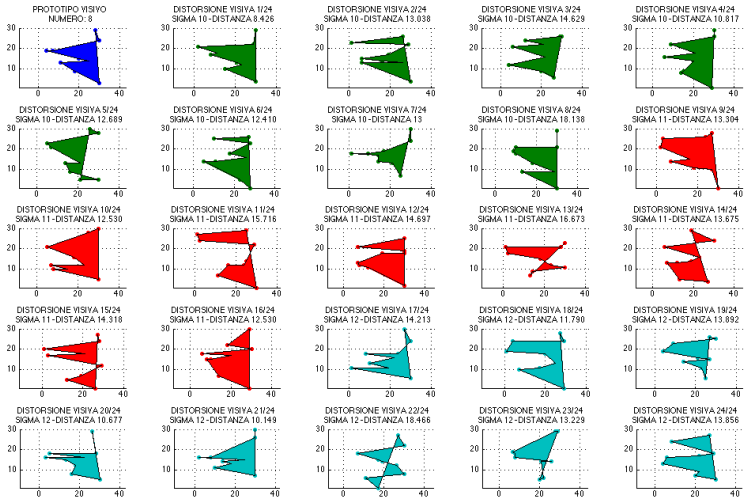

Figure 11: A visual pattern used as prototype and its distortions

The green ones are generated with smaller sigma distortion value and are more similar to the original prototype showed in the left upper corner in blue. The red ones and light blue ones are more distorted, according to higher sigma values used during generation. We want to emphasize that the display itself is subject to approximation: in fact, `scatter` and `patch` functions used to transform the points in figures are very sensitive to coordinates variations. Then a stronger variation on multiple points can result in a different order in which the points are taken into consideration to close the line that defines the contours of the figure, generating in conclusion a pattern that appears to be diametrically different from the original one. If we analyze the coordinate values, these do not differ more than anticipated by those of the prototype of departure.

- Test to see how distortions are far (on average) from respective prototypes. This test-suite include analysis of distortions generated from the prototype, distance of the distortion generated by the nearest prototype (and farthest) to the prototype, the average of the distances of distortions, generated by the prototype, from the prototype and the average distance, normalized to the average maximum distant distortion, generated by the prototype, from the prototype itself.

- Test to see how the samples of each prototype are distorted with each other at different levels of sigma.

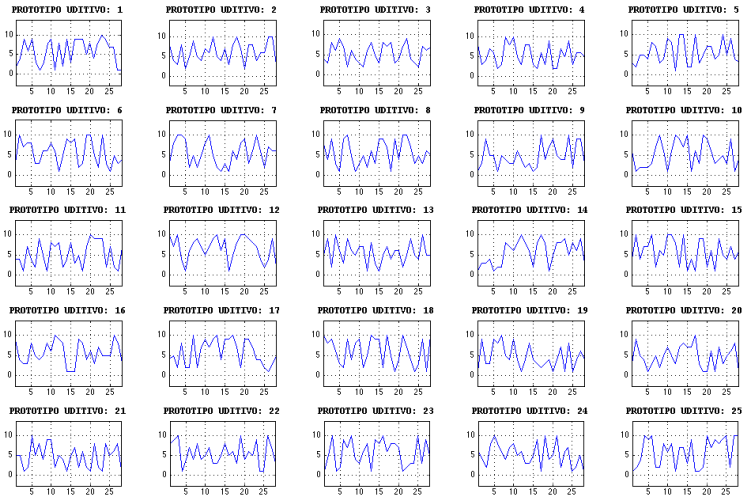

Figure 12: Twenty-five auditory patterns used as prototypes

From each of them, as for visual prototype patterns, have been generated 24 distortions, using different Gaussian sigma values: the pattern are shown in sinusoidal grid by MATLAB.

A desirable feature is to have distortions sufficiently distinct to define a category plausibly made of multiple exemplars but, at the same time, such as to allow map to categorize them in the same area. Distortions too different from each other and especially from initial prototype, lead the map in the training phase to get confused in finding common characteristics; on the contrary, too similar distortions would lead the map to obtain the best results, but that would neutralize the next Hebbian training/testing because there are not categories in the input space, but minor variations of the same patterns (i.e. single dots, no real similar exemplars of same kinds).

3.2.4 *SOMs: training and testing*

During the training and testing of the maps (still disconnected) we tried to find the right balance between learning parameters and desired results: as mentioned above for input patterns, in order to give concrete meaning to the results of the experiments conducted thereafter, it was

necessary to study the conformation of the categories on the map as a result of numerous attempts to training. One of the most desired features is to get a map that, at the end of training, assigns to each neuron a different prototype: the individual prototype's distortions should be assigned to the same neuron (in small quantities) or neighboring neurons. This result is optimal because it allows us to identify precise category boundaries but, at the same time, diversity within the exemplars that define a category, with the guarantee that categories do not overlap, except for "marginal" (compared to the boundaries of the respective category) exemplar. In order to achieve this effect, an automatic test has been prepared, capable to try various configurations of learning with specific tests conducted to obtain the desired paramaters: adjust the parameters of a SOM is a work of art!

To test the SOMs was prepared a test-suite that includes:

- Test to see the performance of the map in terms of global Q error.
- Test to study the activation of the units of the map on prototypes, which shows the BMU and the Q error for each prototype (remember that the prototypes are not included in the training, they are used only once to generate copies) and the number of unique BMU: if it is equal to the number of prototypes, the map assigns to each prototype of each category a distinct neuron.
- A table of activation of the units on the map. Given a row k and a column k of the table, there are shown the number of generated exemplars starting from the prototype r such that the BMU that is activated is at a distance y from the BMU of the prototype r. In Figure 13 we see a screenshot showing this test (useful for understanding how the map forms categories) and test maps to find the right amount of noise (this was a fundamental task in the first phase of implementation).

3.2.5 *Visualization tools*

Many of the analyzes during debugging and testing have been completed through the use of various visualization tools. For example, we

Distanze BMUs:	0	1	2	3	4	5	6	> 6
Vis. Prot 1	10	14	0	0	0	0	0	0
Vis. Prot 2	18	6	0	0	0	0	0	0
Vis. Prot 3	23	1	0	0	0	0	0	0
Vis. Prot 4	12	12	0	0	0	0	0	0
Vis. Prot 5	24	0	0	0	0	0	0	0
Vis. Prot 6	15	9	0	0	0	0	0	0
Vis. Prot 7	21	3	0	0	0	0	0	0
Vis. Prot 8	14	10	0	0	0	0	0	0
Vis. Prot 9	17	7	0	0	0	0	0	0
Vis. Prot 10	22	2	0	0	0	0	0	0
Vis. Prot 11	24	0	0	0	0	0	0	0
Vis. Prot 12	24	0	0	0	0	0	0	0
Vis. Prot 13	21	3	0	0	0	0	0	0
Vis. Prot 14	17	7	0	0	0	0	0	0
Vis. Prot 15	15	9	0	0	0	0	0	0
Vis. Prot 16	24	0	0	0	0	0	0	0
Medie tot. #:	18	5	0	0	0	0	0	0
Medie tot. %:	75%	22%	1%	1%	1%	1%	1%	2%

Figure 13: SOM testing activation table

The image was cut for space reasons: in the original screenshot, there were 100 lines (i.e. 100 prototypes). Each column shows a header (0, 1, 2, 3, 4, 5, 6, > 6) that represents the distance (in topological terms) between units on the map. So on line 1, we see information about activation of prototype "1" distortions. Say that prototype "1" has BMU x. Than, if presented to the map, its distortions will activate some units. From line 1 we know that 14 of them activate the same unit x, and 10 activate units which are at distance 1 from x in the lattice of the map. The final line was cut and pasted with line showing average and a percentage the number of units that activate the unit prototype and those that turn around prototype.

have made extensive use of $\mathrm{som}_{v}\mathrm{oronoi}$ function, an "ad hoc" method that recreates the Voronoi diagram of the space defined by weight vectors of the units and displays each category boundary with a different color on the topologically organized lattice. This allowed us to study the evolution of the boundaries of the categories during training: moreover, the function is parametric and can be used to color the map with a predefined number of colors, so as to assign the same color to each category and study how new categories, introduced during the same training, are topologically organized. We will explore the experiment using training with incremental introduction of the categories in the final pages of this chapter and in detail in chapter 4. In the next figures are shown some

screenshots of tools described above. Figure 14 shows, from left to right,

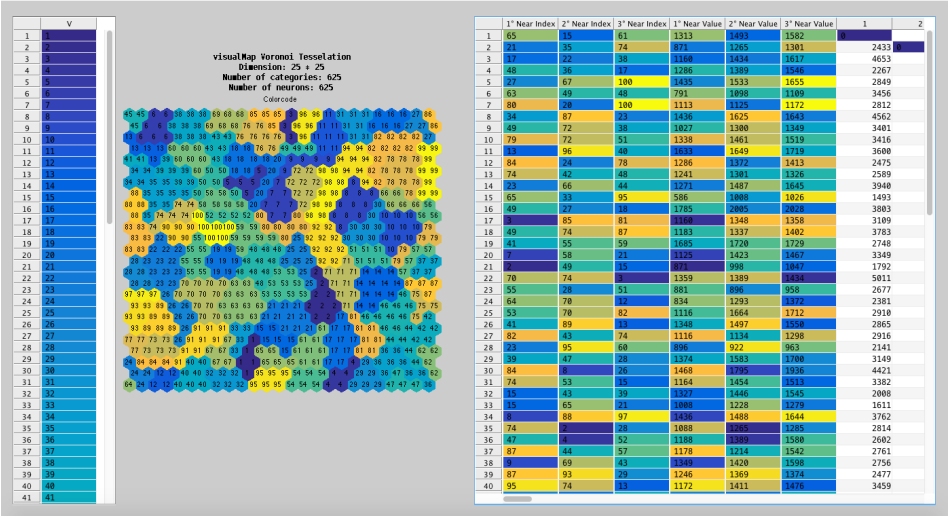

Figure 14: Auditory SOM Voronoi tesselation for 100 categories.

one column with 100 prototypes from which distortions were generated, with which the map has been trained: to each of them has been assigned a color. The colors are not willing "in an orderly manner" because, despite the "names" of prototypes (categories, $1, 2, .., 100$) are ordered, the prototype 2 is no (necessarily) more "similar" to the prototype 1 than it is the prototype 80. Subsequently, the map is displayed: each neuron is colored with the color assigned to the prototype for which the distance "prototype-weight vector" is minimal. In addition, within each unit is inserted the name of the prototype that is activated. That means, basically, distortions of category 45 (generated from prototype 45), will be recognized as exemplars of 45 from 3 neurons in the top left corner of the map. Some categories have boundaries that delineate a greater number of neurons than other ones: the origin of this behavior lies in the way in which the distortions used in training are distributed, in the order in which they are drawn, and more generally depend from the training of the map. It's hard to investigate the exact causes, but we can imagine that the category 63 has been formed following the presentation of distortions more "distant" between each other or, again, quite similar to those used to define the nearby categories. On the right we find a table showing the relative Euclidean distances between the categories in the map - with a highlight on the three nearest categories. The picture above, Figure 15,

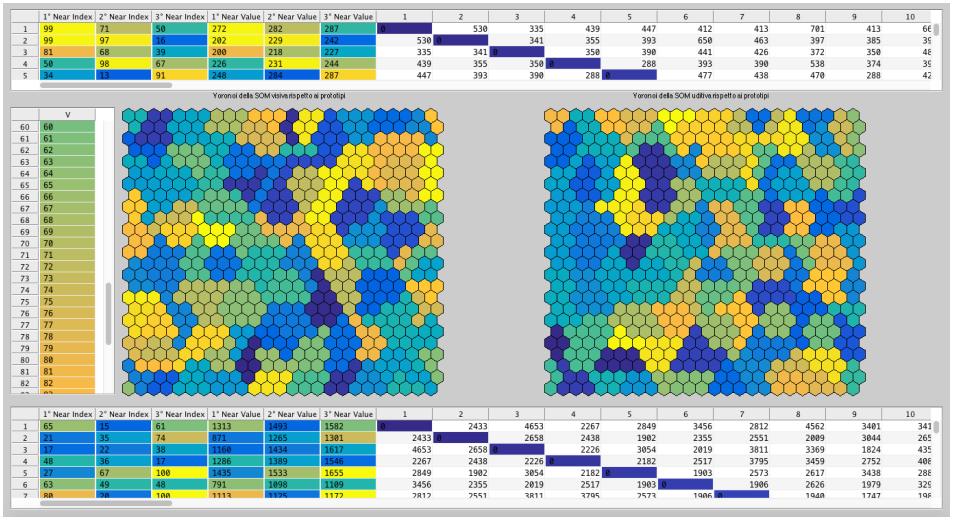

Figure 15: Visual and auditory SOM Voronoi tesselation for 100 categories.

shows a view equivalent to that shown previously, with a direct focus on both maps simultaneously. This view was particularly useful for the study of the behavior of Hebbian training (or rather, the consequences of the conformation of the maps on Hebbian training). An additional instrument was designed to debug and test the Hebbian training, in which you can put the focus on the changes that affect a single category: more on this and some images are shown in the next section.

3.3 HEBB CONNECTION AND LEARNING BY ASSOCIATION

3.3.1 *Introduction to Hebb learning*

Hebbian connections are crucial for the functioning of the model. Indeed, they allow you to connect the two maps and embody learning by association. Before explaining in detail the way in which these connections between maps are created, strengthened and used to get a taxonomic response and the ability to fast-mapping mentioned previously, we introduce the meaning they have in the model and how this fits with what happens in a biological brain. From the point of view of artificial neurons and artificial neural networks, Hebb's principle can be described as a method of determining how to alter the weights between model neurons. The weight between two neurons increases if the two neurons activate

simultaneously, and reduces if they activate separately. As Donald Hebb says in his most influential work ([7]), *The Organization of Behaviour: A Neuropsychological Theory*:

> «when one cell repeatedly assists in firing another, the axon of the first cell develops synaptic knobs (or enlarges them if they already exist) in contact with the soma of the second cell.»

This rule, also known as Hebb rule, is often paraphrased as «neurons that fire together wire together». The American psychologist Gordon Allport posits ([1]) additional ideas regarding cell assembly theory[11] and its role in forming engrams[12], along the lines of the concept of auto-association, described in [1] as follows:

> «If the inputs to a system cause the same pattern of activity to occur repeatedly, the set of active elements constituting that pattern will become increasingly strongly inter-associated. That is, each element will tend to turn on every other element and (with negative weights) to turn off the elements that do not form part of the pattern. To put it another way, the pattern as a whole will become 'auto-associated'. We may call a learned (auto-associated) pattern an engram.»

It is easy to see how both the Hebb rule and the consideration proposed by Allport on the learned auto-associated pattern leads to the explanation of the formation of an associative memory that fits well with the initial idea to get a more or less faithful simulation of the cognitive process that binds the words to the categories (in this context, visual, but potentially recognized by any other human sense) that the individual is able to distinguish. In this scenario, the two SOMs are sets of *neurons* that are active *simultaneously*[13], the Hebbian connections bewtween play the role of *synapses* connecting the two *cortices* (i.e. simulated by SOMs) and the patterns mentioned by Allport are the memories of associations created

11 We talk about it in subsection 2.1.2.

12 An engram is a hypothetical neurobiological element that allow the memory to remember facts and feelings, storing them as biophysical or biochemical changes in the tissue of the brain and other nerve structures.

13 During a labeling event, see below in text.

in all past event, which reinforce or weaken with the experience, according to the theory that memory resides precisely in the form of *engrams*, or biophysical or biochemical changes in the brain.

We present all the *organic ingredients* needed to simulate the process of Hebbian associations. In the next pages we present the process of synaptogenesis that occurs in the brain and the mathematical formulation of the model.

3.3.2 *The complex synapto-genesis process*

«Randomly formed synapses appear to form a substrate for the development of circuits that are dependent on environmental input. The ready availability of billions of unspecified synaptic contacts in the immature cerebral cortex may be important for the formation of the synaptic circuits underlying the development of higher cortical functions, including mathematical skills, musical ability, and language functions.» Peter R. Huttenlocher, [9]. In opposition to what was done[14] by Mayor and Plunkett in [21], we introduced random synapses at once (although we conducted tests with incremental introduction) in our main demo to simplify the process and to study without too much noise the evolution of Hebbian connections between the two maps.

Synaptic pruning Another neurobiological process that occurs in real brain is the synaptic pruning: the number of synapses decreases by a process of elimination of weak synapses. This mechanism is driven by experience and is thought to minimize energy consumption. As we will emphasize, this aspect together with the incremental synapto-genesis affects the operation and performance of the model, but also introduces an additional tuning parameter that makes it even more complex to perform the tests. «Any modelling endeavour that attempts to replicate aspects of human behaviour is inherently committed to a set of simplifying as-

14 They model "blind" synapto-genesis, the process of forming synapses at random locations by linking together only a percentage of randomly picked neurons on both maps, a percentage that increases linearly with age, from 1% connectivity to full connectivity after 500 epochs of training.

sumptions. This commitment is a double-edged sword. On the one hand, simplification offers parsimony: the fewer assumptions required to account for the data, the better the explanation offered by the model. On the other hand, simplification typically involves highlighting some aspects of behaviour at the expense of others. Furthermore, simplification can sometimes trivialise explanation.» - Mayor & Plunkett, [21]. Our ultimate conclusion on the efficiency of association learning is strongly linked to specific considerations on the training and performance of the maps. As pointed out also by the authors of the model, this aspect is crucial for the proper formation of solid Hebbian connections and the attainment of a sufficiently extensive and precise. In conclusion, despite this process of pruning is fundamental and our realization of the model also provides for the possibility to simulate this cognitive process (according to tests conducted by Mayor and Plunkett), to realize quickly and in the time at our disposal the dynamic expansion of the maps, the new features proposed presented in chapter 4, we missed the appearance of synaptic pruning, and we focused on targeted testing to study the progress of the Hebbian training under fixed conditions.

3.3.3 *Implementation and training algorithm*

FORMALIZATION The Hebbian connections are simulated through an $N \cdot M (= S)$ matrix where N and M respectively correspond to the number of neurons of the visual and auditory map and S to the number of synapses[15]. To simulate the process of synapse genesis, all S synapses between the maps were first randomly initialized with a normal distribution centered on $\frac{1}{\sqrt{S}}$ and with a standard deviation of $\frac{1}{\sqrt{1000 \cdot S}}$. Synapse amplitudes are modulated according to a standard Hebb rule with saturation. Therefore, synapse weights stay in a physiological range even for high neural activities. The synapse connecting neuron i from the visual map to neuron j of the auditory map is computed as follows:

$$w_{ij}(n+1) = w_{ij}(n) + 1 - e^{-\lambda \cdot a_i \cdot a_j} \tag{9}$$

15 The whole process follows the same formalization proposal in [21].

where n refers to the index of the word-object pairing and $\lambda(= 10)$ is the Hebbian training learning rate[16] and a_i and a_j are the *neural activity* of neuron i and j on input pattern $n + 1$. We define the neural activity of a neuron k to be:

$$a_k = e^{-\frac{q_k}{\tau}} \tag{10}$$

where q_k is the quantization error (presented in subsection 2.2.2) associated with neuron k and $\tau = 0.5$ is a normalization constant. After every word-object presentation (during labelling events), weights are normalized:

$$\sum_{ij} w_{ij}^2 = 1 \tag{11}$$

so as to model limited synaptic resources.

TRAINING ALGORITHM The model first learns to categorize objects or labels in an unsupervised fashion. Through unimodal presentations, the maps self-organize and form categories out of a complex input space. All category boundaries are defined during the unsupervised learning phase. Object categorization and word-form categorization are thereby determined during unsupervised learning activities in the model. In the supervised phase of learning, single cross-modal labelling events need only to bind the categories on each map. The joint attentional events themselves provide the supervision needed in order to generalize accurately word-object associations. Word learning therefore corresponds to a supervised, cross-modal learning activity in the model. Joint attentional events are mimicked through the simultaneous presentation of objects and their labels (one pairs for each categories defined earlier) and constitute the supervised component of word learning that is essential for learning the arbitrary mappings between labels and objects. Synapses connecting active neurons on both maps are reinforced through Hebbian learning, using Equation 9 and depending on Equation 10. The more maps are organized in a consistent way, the lower is the quantization error (q_e) accumulated on the matching units that actually respond with greater intensity to the input pattern presented in (belonging to, say, the category X): this, in case

16 Different from SOM learning rate.

the neuron i is an attractor of pattern belonging to the category X, leads to reduce the coefficient of the Equation 10, which is negative, then results in a more intense activation. This process makes sense because a lower degree of uncertainty implies a stronger activation of the neuron. Conversely, for neurons h that are not attractors of patterns belonging to category X, the values of quantization error (q_e) will be greater leading to a lower activation of the neuron and to a consequent minimum (or zero) weight increase of the respective connections with the neurons of the auditory map.

Thus, due to the topographical organization of the maps that takes place in early development, many neighboring neurons on each map will be activated by the presentation of an object and its corresponding sound pattern. Cross-modal Hebbian learning will then take place for neighboring neurons on each map, reinforcing the connections according to intensity of activation of the various pairs of neurons, intensity linked to the degree of uncertainty of the neuron and consequently the overall efficiency of the two maps. Therefore, the association between the paired object and its corresponding sound pattern will be generalized, automatically building associations between all objects in its category to all sound patterns of the appropriate type. A single labelling event is thereby able to induce a taxonomic response with the label extended to all objects of like type: the novel word is learnt.

After training on cross-modal pairings, we assess the capacity of the network to extend the association of a presented word-object pair to non-paired items that belong to the same category.

3.4 HEBBIAN TESTING

3.4.1 *Definitions*

In order to assess the performance of the model, it is necessary to introduce two definitions. The first definition introduces a measure of the taxonomic response. It may be interpreted as the learnability of a word. The second defines a criterion for deciding when a word is learnt. All results refer to these measures.

Taxonomic factor The taxonomic factor is the percentage of correct word-object associations, following one single labelling event per category. So it also approximates the likelihood that a word is learnt following a single labelling event.

Learned word We will consider a word to be part of the lexicon if more than 80% of the objects of the appropriate category are associated with the correct label.

The authors conducted many tests, the first of which aimed to show how the organization of the maps play a vital role in the performance of the model obtained after Hebbian learning. For example, on pag. 21 [21], it is shown a graph of the trend of taxonomic factor obtained after Hebbian training (one single labeling event in category) when this is introduced at different points of the SOMs training.

We did many tests, the first of which aimed to show how the strength of the maps play a vital role in the performance of the model obtained after Hebbian learning. In chapter 4, we will show result of this experiment, conducted using the same connections for each Hebbian training introduced and showing that old Hebbian connections, developed in the early stages of SOM training, do not affect the performance of learning introduced successively on the same maps in which the training is continued until reaching a better quality (with minor global Quantization Error because of multiple epochs of training).

3.4.2 *Errors' evaluation*

«Although the network is able to generalize single label-object associations in a taxonomic fashion after object and label categories are well-formed on the maps, the same cannot be said for the network during earlier stages of training. After limited exposure to the visual and object environments, the network is prone to generalize newly acquired label-object associations in an inappropriate manner. For example, labels may be used to identify objects outside of the appropriate category» - Mayor & Plunkett, [21]. One of the key aspects of Mayor & Plunkett leverage re-

garding the performance of their model is linked to the results obtained in the first phase of training, the unsupervised one, in which the two visual-auditory models are educated. To identify errors of categorization subsequently made by the model, two types of events are introduced: a **production event** when a pattern of activation occurs on the auditory map as result of the presentation of a visual object to the visual map and subsequent propagation of activity through the Hebbian connections to the auditory map. A **comprehension event** occurs when a word is presented to the auditory map resulting in a pattern of activation on the visual map via the cross-modal Hebbian connections. Therefore, there are two possible tests: production tests test the reliability of the model in terms of the ability to name the object with the right end shown, tests in comprehension test the ability to *indicate* the right object after listening to a given word. We emphasize indicate because the respective experiment conducted in the laboratory with real children requires the presence of 2 or 3 objects in a restricted visual scene and the children indicate the object that is suggested (using voice) from caregivers. In this scenario, the model does not have a set of possibile choices: what is going to test is the visual representation of the model recreates for the object of which it *hears* the name. However, this makes sense, because it is assumed that the child, to recognize the object in the visual scene, must somehow recreate an arbitrarily complex object form in his visual cortex and compare it with the forms that he identifies in the visual scene. For the authors, overextension errors arise from confusions on the visual map. Confusions on the auditory map can be interpreted as insensitivity to mispronunciations of a word (in production, model pronounces "doll" after seeing a dog image because it confuses pronunciations of the two words) or slips of the tongue (in comprehension, hearing the word "doll" might lead the model to incorrectly identify the object dog for the same reason). However, during the testing phase, to more accurately isolate faults, we focused on the production capacity of the model and have identified a fallback for those identified by the authors. In the next section we introduce our version of the production-testing algorithm. We would like to emphasize the importance of this algorithm processed to evaluate the performance of the Hebbian training: the elaboration of possible cases that can occur during

testing of Hebbian model was the result of an intensive analysis, which allowed us to isolate in detail the causes of the errors committed by the model and, consequently, to consider real causes of limitations and try to overcome them with the experiments presented in chapter 4.

3.4.3 *Production testing: a complex algorithm*

To understand the algorithm, we must first list some of the conditions necessary to detect possible cases. Before introducing the pseudo code (in Algorithm 3), we go to identify all the possible cases in term of "results" obtained. Precisely, this test plans to present a visual object on the map, and see what corresponding word is activated on the map auditory, choosing as induced BMU the unit such that the Hebbian connection, starting from the BMU on visual map, is more intense. As already pointed out, the errors that the model can make in this scenario are different: the error may affect the choice of the unit on the visual map (error in image recognition / categorization), the choice of the induced BMU on the auditory map (error due to the intensity of the Hebbian connections) or both cases. We might initially think that the three cases are the only possible ones, however, this analysis does not allow us to distinguish between errors of over-extension and misinterpretation: however, the combination of these errors may lead to *assess as correct* a specific production test on a presented object, despite this being incorrectly recognized (in visual map). Suppose we present to our model a visual distortion of a dog picture: it may be that the model recognizes incorrectly the image as an artifact (i.e., a house) and that, simultaneously, choose as induced BMU (always incorrectly, due to the strength of the connections Hebbian) a BMU in auditory map that identifies the word dog. In this circumstance, if we simply look at the result on the auditory map, we would say that the test production has been successful, despite the model has made two serious errors!

In fact, the first error lies in having recognized a dog recreating the representation of a house (it is assumed that the visual encoding of the artifact house is very different from the visual encoding of a dog), a sign that the visual map is not well self-organized, and then connected the house to the name dog, a sign that the Hebbian training was not carried

out correctly. A similar situation may relate to the showing of a dog and its visual recognition in a house, with pronunciation of the word "doll". Again, we have a (big) visual recognition error, that would be accepted as an over-extension if we looked at only the result obtained on the auditory map. According to the examples above, we give some formal definitions of the errors and then we illustrate the algorithm of production test that analyzes all possible scenarios.

3.4.3.1 *Definitions*

PROTOTYPICAL BMU When we present a pattern to the visual map, previously self-organized, a BMU is identified on the map (see section 2.3). According to the topological organization of the map, we can identify, for each category c, a *prototypical BMU*, which is the unit that is mostly activated if we present to the map the prototype from which was generated all exemplars of the given category c. We call a prototypical best-matching unit BMUP (*P* stands for "prototypical"). For example, suppose we have created a prototype *Dog*, from which we have generated 12 distortions. Subsequently, we trained the map in such a way that recognized exemplars of dogs and other animals or artifacts. As shown in subsection 3.2.5, we can define a Voronoi tessellation on the map, such that a given number of units will activate for (at least) as many separate exemplars of dogs seen during the training phase. In an ideal situation the category boundaries are well defined, and we can imagine that the prototype pattern of *Dog* is recognized from the most central unit in the *neuron-area* that recognizes the *dogs*. If we look at the upper part in the Figure 16, we see a map trained using some distortions of dog, specifically those in the marginal range shown on the left. Subsequently, the prototype pattern *Dog*, or the dog at the center of the polygon shown on the left from which all the distortions were created, was presented to the map. The map has responded by providing the BMUP, the colored blue in the center surrounded by rose. The category *Dog* recognized by the map is given by all the BMUs such that the respective weight vector is maximally close to the BMUP. The topological arrangement of neurons that recognize exemplars of a given category is not always symmetric and well distributed around the prototypical unit: the borders of a category depends substantially on how

training takes place on the map, the order of presentation of the distortions, the degree of distortion of the same and those of other categories. However, to determine whether the map properly recognizing a generic dog, it is not necessary to verify that the pattern presented actives exactly the BMUP of category *Dog* (call it BMUP_{Dog}): given x the BMU activated by specific dog-pattern, it's enough to check that the BMUP[17] closest to x is equal to the BMUP_{Dog}, as shown in the middle of the Figure 16.

17 Taken from the set of all BMUPs for all categories.

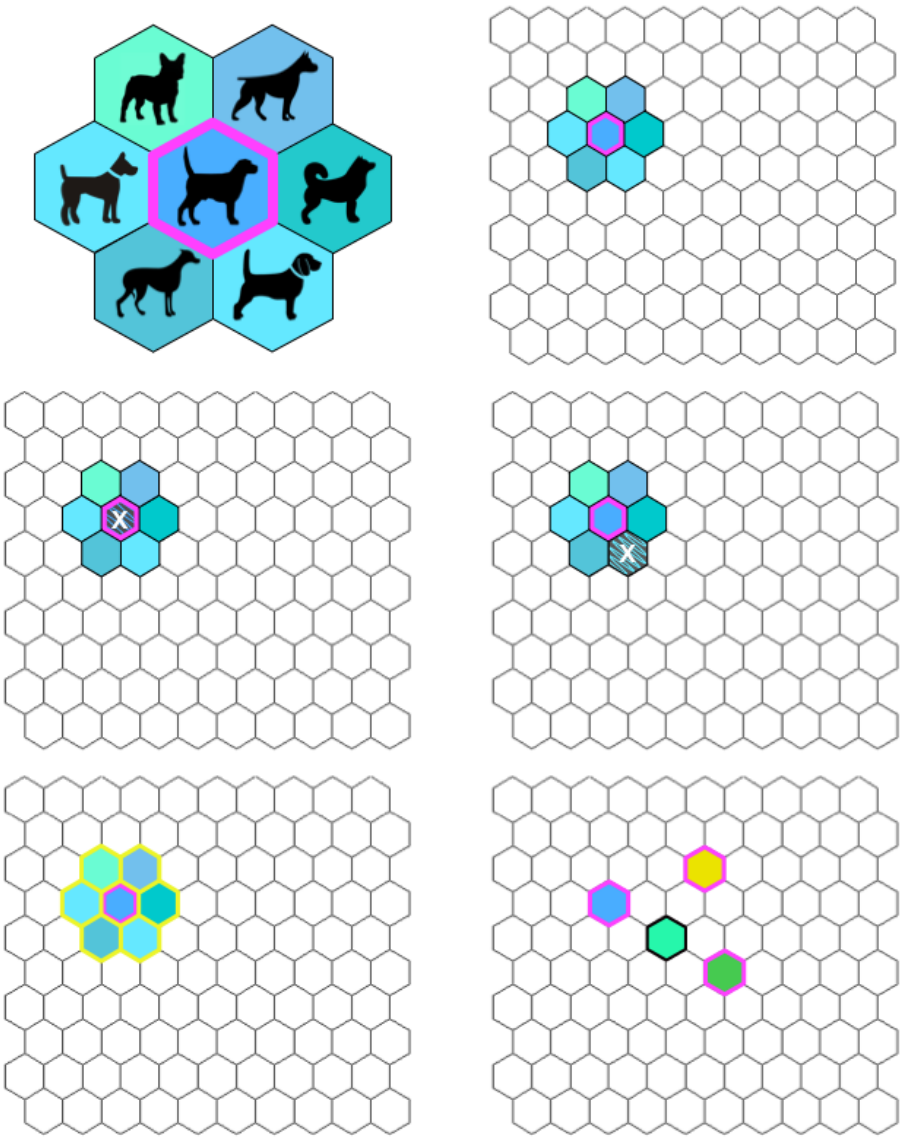

Figure 16: An explanation about prototypical BMU on visual SOM.

In the upper part, we see category of Dog optimally formed on the map. In the middle, we see the two scenario in which the right BMU is active: from left to right, the BMU is equivalent to the BMUP and part of the set of BMUs of category. In the lower part on the left, we see the prototypical BMUP Dog surrounded by pink, and around the BMU identifying dogs surrounded by yellow. On the right, we see three prototypical units surrounded by pink. At the center, a greenish BMU. When an exemplar of dog is presented to the map and the greenish BMU is activated, if we consider the prototypical BMUP closer to the greenish unity, these are three.

In the lower left corner are shown all correct BMUs: we could say that the map correctly recognize the visual pattern x of a dog if and only if the BMU of pattern x correspond exactly to the $BMUP_{dog}$ (surrounded by rose, the left case in the middle) or is part of the set of BMU recognizing exemplars of that category (surrounded by yellow, the right case in the middle). Formally, we say that a pattern is properly defined by a map using the following formula:

$$map\_result_{strict}(x) = \begin{cases} Correct & BMUP_{closest} == BMUP_{real} \\ Incorrect & otherwise \end{cases}$$

with x is the pattern, $BMUP_{closest}$ is the closest prototypical BMUP (eventually exactly $BMUP_{real}$), and $BMUP_{real}$ is the real prototypical BMU of the category which x belongs to, i.e. $BMUP_{dog}$.
However, we could be more lax and verify that the real prototypical $BMUP_{Dog}$ is part of a set defined by units in a given range of prototypical BMUPs closer to activated BMU of a given pattern x: in lower right corner, we see that $BMUP_{Dog}$ is included in the set of neighbors categories ($BMUPs_{real}$) in a given range from the central greenish BMU activated by pattern x. Among them there is also the prototypical $BMUP_{Dog}$ desired. In this case, we suppose that the map has correctly recognized the exemplary because the respective BMU is equidistant from several prototypical BMUPs, but is still close to the boundaries of the correct category, i.e. *Dog*. The topological radius within which we consider the prototypical units is limited to 2 in our implementation.

Formally, following this lax definition we say that a pattern is properly defined by a map using the formula:

$$map\_result_{lax}(x) = \begin{cases} Correct & BMUP_{real} \in BMUPs_{closest} \\ Incorrect & otherwise \end{cases}$$

with x is the pattern, $BMUPs_{closest}$ are the set of closest prototypical BMUP in a fixed topological range from BMU activated by pattern x, and $BMUP_{real}$ is the real prototypical BMU of the category which x belongs to, i.e. $BMUP_{dog}$.

Weaker definition expect to consider not only the best-matching unit BMU after presentation of pattern x, but also the second and the third

units maximally active to build the sets of neighbors, allowing flexibility in categorization task of the map. In the next paragraph, we introduce the concept of induced BMU.

HEBBIAN CONNECTION AND INDUCED BMU The matrix of Hebbian connections (HM) between the two maps is organized as follows: given N and M neurons on the visual and auditory SOM respectively, the matrix has dimension $N \cdot M$ and in the position i, j we find the intensity of the synaptic connection between the neuron i on visual map and neuron j on the auditory map[18]. Fixed a neuron i on the visual map, we define the induced neuron j on the auditory map such that $HM(i, j)\ \forall j$ is maximum. Moreover, fixed a visual pattern x, if the neuron j is the induced auditory neuron from the BMU in visual map, then we call the neuron j - IMU (Induced Matching Unit) and the BMUP[19], from the neuron j - IMUP. During a production test, as already said, a visual pattern occurs and that the neuron induced on the auditory map recognizes the right label associated with the visual pattern. We can then define these conditions[20] useful to understand the algorithm presented later:

- $a\_condition$: $BMUVP \equiv BMUVP_{real}$
- $b\_condition$: $\neg a\_condition \wedge dist(BMUVP, BMUVP_{real}) <= \epsilon$
- $c\_condition$: $IMUAP \equiv BMUAP_{real}$
- $d\_condition$: $\neg c\_condition \wedge dist(IMUAP, BMUAP_{real}) <= \epsilon$
- $e\_condition$: $\overline{IMUAP} \equiv \overline{BMUAP}_{real}$

The $a\_condition$ is $true$ if the result of one of the implementation of $map\_result(x)$ presented above is "Correct" for pattern x. The $b\_condition$ is $true$ if BMUVP calculated from the BMU of pattern x is different from $BMUVP_{real}$ but the topological distance between BMUVP calculated from the BMU of pattern x and $BMUVP_{real}$ is less than ϵ.

18 The numbering of neurons in each map follows a default order, independent from the topological two-dimensional coordinates.

19 Calculated with definition given in the previous paragraph.

20 *V* in names is for "Visual" and *A* for "Auditory".

For c_condition and d_condition are worth the same conditions: the only difference is that formulas use as starting IMU and IMUP, on the auditory map, respectively the one induced from the BMU activated on the visual map after pattern presentation and the prototypical auditory one calculated as say above. The last condition, e_condition, is similar to c_condition: however, it refers not to the real (right) category which the pattern belongs to, but to the category which it belongs to *according to the map result*. That's why the there is an over-line on IMUAP: it refers to the prototype defined by the *right or eventually wrong [but similar to the correct-one]* category assigned to the pattern. For example, when a *dog* is shown on the map, and it is categorized as a *cat*, the c_condition is true if the BMU activated on the map is associated with the word *dog*, the e_condition if the BMU activated on the map is associated with the word *cat*.

Indexes

Regarding the above conditions and the following algorithm, some variants have been developed taking into account the first x Matching-Unit more active following the presentation of the visual pattern and / or by examining the first y Hebbian connections stronger than the others. Something about these variants was already said in the subsubsection 3.4.3.1, during presentation of prototypical BMU: we can say that the algorithm remains unchanged, and it is possible to build four separate indices evaluation. The first of them is the one that identifies the statistics by exploiting the conditions taken into consideration (the most restrictive). The second looks at the first 3 BMU maximally active at the presentation of a pattern: the conditions could easily be modified and extended transforming some equivalencies in inclusions checks from set theory. The third index looks only at the BMU but treats 3 Hebbian connections stronger. The fourth and final index considers 3 BMU and 3 Hebbian connections stronger for each of them. For each of these four indices, there is the taxonomic response indicator that, in accordance with what is established previously and in a similar manner to what was done by the authors, corresponds to assess whether at least 80% of the pairs presented for a given category are correctly associated (using one of the

four configurations of testing). The four indicators are then expressed as a percentage of the number of categories learned: it goes without saying that the indices less restrictive tends to get a higher value because they are loose in assessing any errors made by the map. Unless specific directions, from now on we will refer to the performance obtained by the process of production Hebbian testing as to the result obtained by making use of the second index (and its indicator expressed as a percentage of the number of categories learned). The process of training and testing can be done in multiple configurations: In this chapter, we presented the foundation through which they were conducted the first tests. We will return to some specific configurations related to the techniques of training and testing Hebbian in chapter 4.
Finally, the production testing algorithm is presented using defined conditions:

Algorithm 3: Production test algorithm

```
Data: Complete Hebbian Testing set
Result: Percentage of classified errors
foreach couple_test ∈ TestSet do
    if a_condition ∧ c_condition then
        corrects += 1;
    else if a_condition ∧ d_condition then
        slip_of_tongue_error += 1;
    else if b_condition ∧ e_condition then
        /* due to categorization error                    */
        over_extension_error += 1;
    else if a_condition ∧ e_condition then
        /* due to association error                       */
        over_extension_error += 1;
    else
        casual_errors += 1;
end
return stats;
```

According to the pseudo code shown, we analyze the possible scenarios in which it can be concluded the test production of an image on the model. Consider the case in which a dog is shown to visual map of the model: we want the model to assign the right label of shown exemplar. In a real scenario, this experiment is equivalent to ask a child the name of the animal that is shown, in this case a dog, as shown in Figure 17.

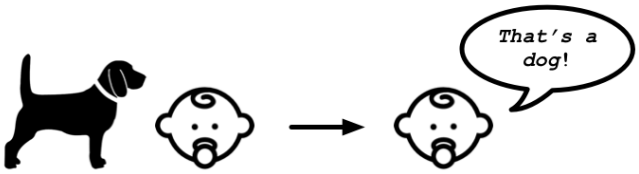

Figure 17: Production test in a real scenario.

The tests conducted on the model allows us to do some extra consideration: in a real scenario we can not *see* the mental representation of the image seen by the baby. In other words, we can not tell whether the image of the dog shown to the child *physically* face the category dogs that the child has: we can understand if this happens only on the response given to us by the child, however, if your child makes a mistake, we are not able to assess directly and analytically if the mistake is due to a categorization error or an Hebbian training error (bad learned associations). However, in our model it is possible to analyze these two errors separately: this is the reason why the test conducted on the model provides more possible cases, in accordance with what said in the beginning of the section.

For ease of viewing, in the next schemes we will not distinguish between the two scenarios of activation of the BMU presented previously: the BMU will always appear as equals to the BMUP, however, the reasoning proposed are also valid in case the BMU activated is part of the set of BMU that are activated by presenting other exemplars of the given category. Particular attention should be paid in Hebbian connection shown: the strongest link are not considered to be the one which link the BMUP to the auditory units, but it's always chosen from the BMU activated by the pattern (which in the drawings is always shown as equivalent to the BMUP, that's why the warning). The two real-world scenarios shown in the Figure 18 are then made to collapse in a more compact display in

which no account is taken of the units that surround the BMUP prototypical.

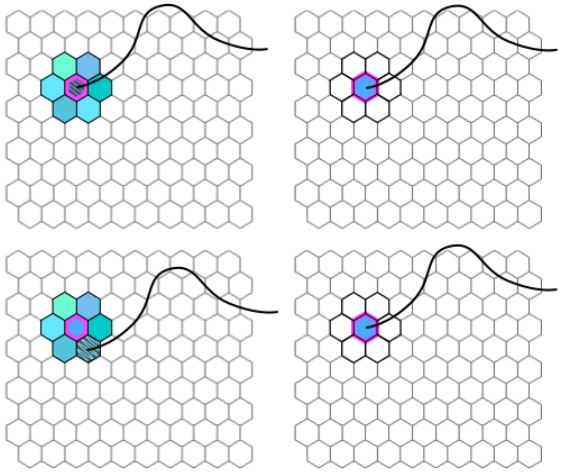

Figure 18: Real BMU activation scenario collapsed in a simple visualization.

In Figure 19 is shown the legend to understand all cases showed after.

Figure 19: Legend for production test examples.

So, we show the most interesting scenarios in which the model does not commit a casual error.

- Case 1: In Figure 20, an exemplar of dog never seen during the training phase is shown to the visual map. One unit, the BMU (call it x), is activated more than others. It is checked that the SOM recognizes the pattern as belonging to the category of the *dogs*, as said above.

Subsequently, the BMU induced k in auditory SOM is selected between M auditory neurons such that:

$$k = max(HM(x, m), m \in 1..M \tag{12}$$

or rather, selecting the one for which there is greatest activity induced (synapses with higher weight). Then it is checked that the prototypical BMU nearest to the one induced (again, see above) recognizes the sound patterns corresponding to the name of shown object, in this case a dog, always according to the technique used on the visual map. In this case, the production test is successful.

Technically, a dog is shown, then *it's seen* as a dog by the model and attached to the label *dog* and *dog* is returned as a label for the object shown. This case is the one in which `a_condition` $\wedge$ `b_condition` is `true`.

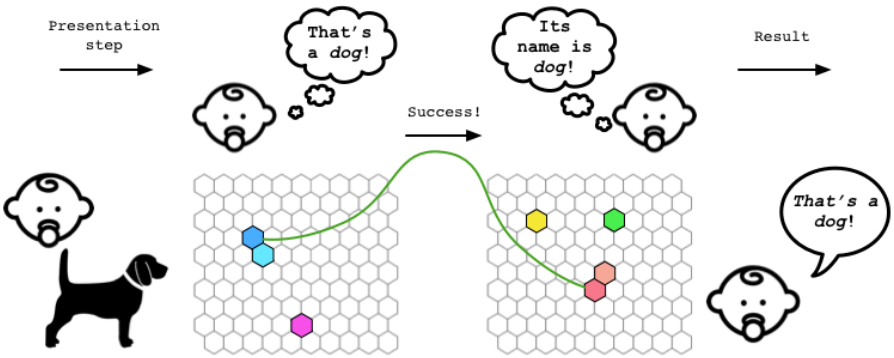

Figure 20: Test production through connections Hebbian: case 1

- Case 2: In Figure 21, an exemplar of dog is shown, then *it's seen* as a dog by the model, attached to the label *doll* and *doll* is returned as a label for the object shown. Technically, a dog is shown, then *it's seen* as a dog by the model and attached to the label *doll* and *doll* is returned as a label for the object shown because the two words *dog* and *doll* are similar: during the Hebbian learning, the connections between the visual units that distinguish dogs and auditory units that distinguish the term *dog* and *doll* were fortified a lot because of the confusion between the two terms on the auditory map. The authors call this type of error due to confusion on the map sound a *slip*

of tongue. This case is the one in which $\texttt{a\_condition} \wedge \texttt{d\_condition}$ is true.

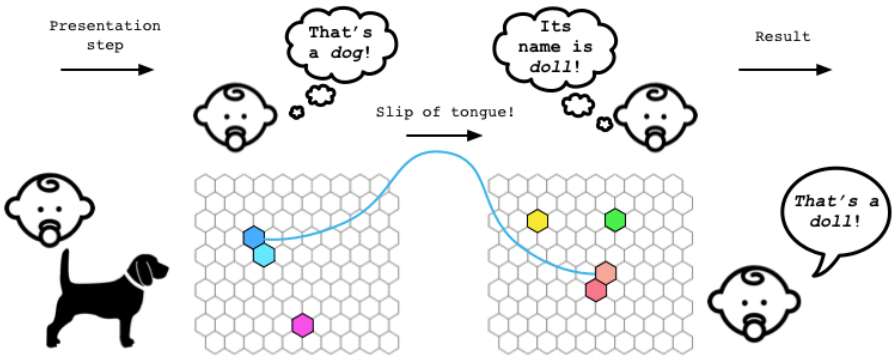

Figure 21: Test production through connections Hebbian: case 2

- Case 3: In Figure 22, an exemplar of dog is shown, then *it's seen* as a dog by the model, attached to the label *cat* and *cat* is returned as a label for the object shown. This case can be discussed: we suppose that the authors of the original model are of the view that this case and the next one are considered cases of over-extension error. Conceptually, if we look at the result on the auditory map, the word *Cat* is over-extended to an exemplar of *dog*. To ensure that this category is actually over-extended, we should study the result of showing a high enough[21] number of exemplars of *cat*. So the association is wrong not because the model confused a *dog* with a *cat* but because it associated to a dog (properly recognized) the label *cat*! It happens that the model makes a mistake of **over-extension due to an Hebbian association error** and not due to an (visual) categorization error. This case is the one in which $\texttt{a\_condition} \wedge \texttt{e\_condition}$ is true.

21 Such that we can say that the model has correctly learned to associate the word *Cat* to the set of *cats*.

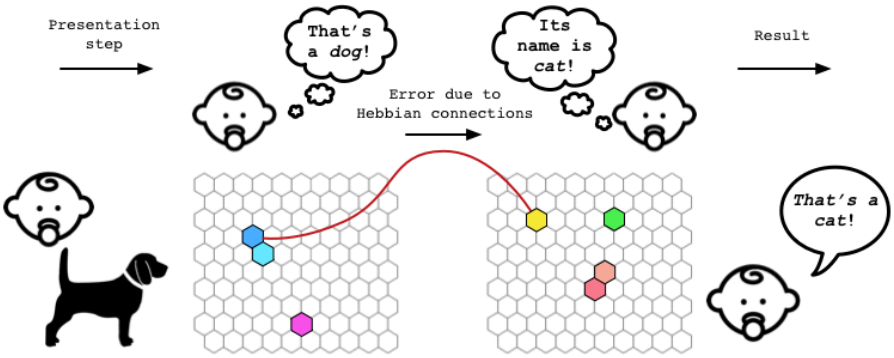

Figure 22: Test production through connections Hebbian: case 4

Distinguish these two cases can allow us to understand what is the major cause of errors of over-extension in the model (error of categorization or association), especially during the experiments with maps evolving: pay attention to the last case we highlight.

- Case 4: In Figure 23, an exemplar of dog is shown, then *it's seen* as a *cat* by the model, attached to the label *cat* and *cat* is returned as a label for the object shown. This is an over-extension error because we have over-extended the category *Cat* over a few examples of *dogs*: in practice, the confusion on the visual map between the two categories results in an error in the accomplished association, and an exemplar of *dog*, actually perceived as a *cat*, is confused for a *cat*. The association is right, but the visual categorization is wrong. In this case, it happens that the model makes a mistake of **over-extension due to a visual categorization error** and not due to an Hebbian association error. This case is the one in which $\texttt{b\_condition} \wedge \texttt{e\_condition}$ is $\texttt{true}$.

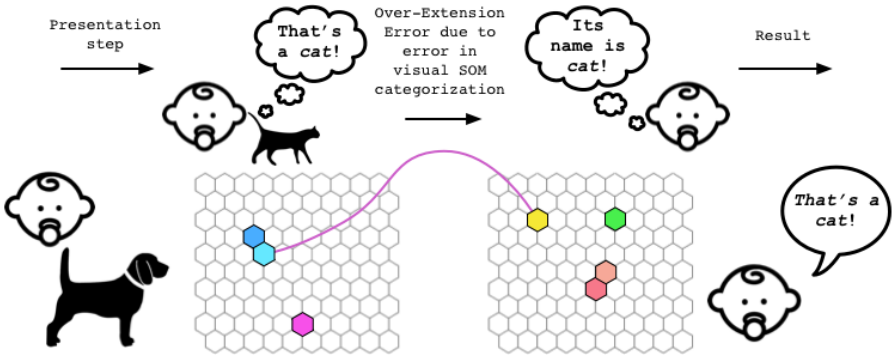

Figure 23: Test production through connections Hebbian: case 8

All the remaining scenarios are considered as casual error (more frequent errors in the model, according to our and authors experiments). In the next section we will discuss about the model in action in three distinct demo.

3.5 MODEL IN ACTION: STRENGTHS AND WEAKNESSES

The last section of the chapter is divided into two parts, each of which introduces a specific testing-configuration for the model or demo. In particular, in the first demo (subsection 3.5.1) we show the generation of prototypes, the training of the two maps and the results obtained in both, presenting some of the instruments implemented to see clearly the topological structure of the categories created. The second demo (subsection 3.5.2) show the classical Hebbian training suite with particular attention to the tools prepared to see an increase in matrix weights and the results obtained with well-trained maps.

3.5.1 *Demo 1: Visual and auditory performance*

The first phase of testing consisted in analyzing the performance of the two maps. As mentioned previously, setting the training parameters of the maps is a difficult task and there are no clear rules to follow. Many tests have been made to achieve a sufficiently stable balance to make inter-

esting the results of experiments conducted later with Hebbian training. Before specifying the parameters of training maps, we provide some data on the inputs used for testing and training. Specifically, using the method described in subsection 3.2.1, 100 visual prototypes were created fairly evenly distributed in space input. Some statistics on these prototypes are shown in the Table 1 below.

Stats description	Value
Minimum distance reached between visual prototypes	24.21
Maximum distance reached between visual prototypes	75.78
Average distance reached between visual prototypes	51.68
Pseudo-upper bound[22] of a reachable distance between two prototypes	123.04
Average distance normalized reached	42%

Table 1: Visual prototypes general stats about reciprocal Euclidean distance.

For 100 auditory prototypes, the same statistics are shown in the Table 2 below.

Stats description	Value
Minimum distance reached between auditory prototypes	13.11
Maximum distance reached between auditory prototypes	30.98
Average distance reached between auditory prototypes	21.46
Pseudo-upper bound of a reachable distance between two prototypes	47.62
Average distance normalized reached	45%

Table 2: Auditory prototypes general stats about reciprocal Euclidean distance.

The performance of the map have not been evaluated both in terms of the topological quantization error or the error, rather in terms of the ability to assign to each category a separate unit (BMUP)[23]. The initial task involved a simulation as faithful as possible to the conditions in which Mayor & Plunkett worked: trains a map with the right amount of noise is a complex task. We can numerically provide a simple explanation of

23 The exemplars used during training are associated with some units topologically close to the prototypical BMU.

the problem. Suppose we have a grid of $25 \cdot 25$ neurons in both maps. We focus our attention on one of the two maps. We have 100 prototypes. We would like the map to assign about 100 separate units to each of them (upon completion of training phase using from the distortions, as the prototypes are never used neither in training nor in testing). If the topology makes use of hexagonal lattice, a neuron has on average 6 neighbors: it is not true for neurons at the edges, so we approximate this number to 5. We have a total of 625 neurons, 100 of which are used to define the prototypical units (what we previously called BMUPs). So there remain 525 "unused": the ideal situation is that at least the 6 neighbors of each prototype units are enlisted to identify exemplars (hopefully sufficiently distorted) of the category defined by prototype. So, set 100 units, 500 units have to be enrolled. That leaves "unused" 25 of the 625 total units. Since many random processes are used throughout the entire process, from the generation of prototypes and samples, through the alignment of the initial weights of the maps and the technical and the order in which the samples are extracted in every epochs of training, we understand how difficult it is to study the trend of the training (remember that training is also linked to several parameters arbitrarily defined including neighborhood, learning rate and number of periods of training) in order to correct the shot and get the desired result.

Performances related to the capacity of categorization on exemplars were measured testing the ability to distinguish fairly accurately but still leaving a certain amount of noise on both the visual and auditory map. To calculate this type of statistics, we have been individually examined categories, and maps have been tested in terms of position in the lattice assumed by BMU activated for each exemplars. More details about how was done this type of test is provided in subsection 3.2.3 and in caption of Figure 13. The results obtained with regard to the visual map are shown in in Table 3:

Stats description	0 (≡ BMUP)	1	2
Average number of visual exemplars recognized by unit at distance k	18	5	1
Percentage of visual exemplars recognized by unit at distance k	75%	23%	1%

Table 3: Average visual categorization result and noise in the map.

The same tests conducted on the map auditory reported the results shown in Table 4:

Stats description	0 (≡ BMUP)	1	2
Average number of auditory exemplars recognized by unit at distance k	20	4	0
Percentage of auditory exemplars recognized by unit at distance k	82%	16%	0

Table 4: Average auditory categorization result and noise in the map.

In Figure 24 we see instead the arrangement of BMUP in the visual and auditory map, left to right. This has been trained for 1000 epochs, taking advantage of the sequential training function[24], with the learning and neighborhood parameters specified immediately below the image.

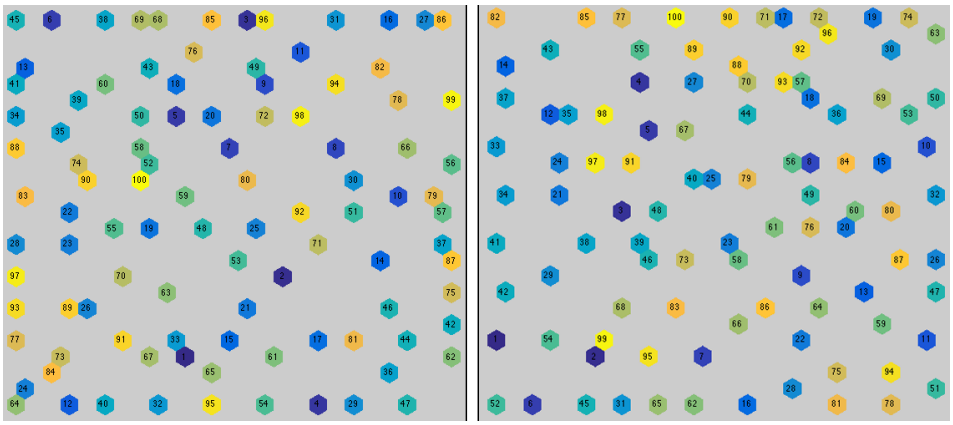

Figure 24: Visual BMUP disposition on visual map.

The learning rate are: $0.8/(1 + (i/200))$. The neighborhood radius are linearly scaled in the range $2 - 1.5$. As regards the auditory map the learning rate are the same, the neighborhood radius varies linearly in range $1.7 - 1.5$. We will not speak of the performance of these maps in terms of QE as we will discuss about this index in the next two demos and in detail the chapter 4, which reported the performance of incremental training and discussion about its correlation with taxonomic factor.

24 `som_seqtrain`, available by the SOM Toolbox - SOM Toolbox

3.5.2 *Demo 2: Hebbian classical training*

Hebbian training standard has been tested first: many exemplars were initially used to train connections in each category, but the most important tests carried out and described in the next chapter, was taken into consideration the strict condition of training that provides a single labeling event with a single word-object couple for each category, as it is the condition that best fits with the emergence of the taxonomic response through fast-mapping, a condition which has been thoroughly discussed in subsection 3.1.2 and more likely that simulates a real situation of association learning. The result, shown in Table 5, was good:

Stats description	Absolute count	% of categories learned[25]
Correct couples using 1 BMU and strongest Hebb connection k	877/1200	70%
Correct couples using 1 BMU and 3 strongest Hebb connection k	947/1200	77%
Correct couples using 3 BMU and strongest Hebb connection k	944/1200	78%
Correct couples using 3 BMU and 3 strongest Hebb connection k	1017/1200	84%

Table 5: Result of production test after 1000 epochs training with 12 exemplars for each categories, and one-shot labelling event (hebb training) for each categories, tested with others 12 exemplars for each categories (different from the ones used for maps training)

3.6 SUMMARY

In this chapter we introduced many fundamental concepts that have marked the entire course of the thesis. First, the generation of the inputs, a task far from simple, in which the graphic tools created for analysis of the performance of the maps (Voronoi tessellation) resulted to be very useful. Second, the presentation and re-implementation of the model, which ultimately results to be full of parameters with vast domains: for this reason and others, the development of the model was a step as interesting as delicate, because we had to exclude from trials all configurations not useful for the achievement of the purpose[26], and debugging the code repeatedly for a long time. Finally, the first complete test of the model, which have brought to light discrepancies in the formal definition of over-extensions

26 Overcome a particular limit, described in chapter 4.

errors during production tests: these have allowed us to study the problem in depth, and implement an efficient and accurate algorithm to test performance during production tests, then used for the second and final part of the thesis. All the work done up to this point has been useful to the understanding of the model and its functioning in its original version. In the next chapter, we will introduce new training techniques designed to address the limitations of the model and the changes made to the model in order to successfully support the new forms of learning introduced: respectively, we are talking about incremental training and the introduction of Growing Self-Organizing Maps as both visual and auditory models.

4

OVERCOMING THE LIMITATIONS

Contents: in this chapter, the evolutionary nature of the model become more pronounced, with introduction of ***incremental trainings*** *conducted to simulate the behavior of the inclusion of new categories, with variation of the* ***dynamic training set*** *when the phase of learning; adaptation of Fritzke expansion technique and* ***introduction of Growing SOM*** *within the original model presented in [21];* ***dynamic introduction of Hebbian synapses*** *as a result of the introduction of new neurons* ***during the Growing SOM training*** *associated with incremental (associative) Hebbian-learning; introduction of development of variants of growth technique, theoretical evaluation of* ***alternative topologies****; analysis of the results of* ***comparison tests*** *between arbitrarily large static maps and maps that grow.*

In this chapter we present the results obtained by some special tests conducted on model to evaluate in detail the causes of the limitations highlighted by the authors and identified by ourselves: next, we will show the techniques that we have developed to overcome these limits, with particular attention to **dynamic expansion** of visual and auditory SOM. In detail, in section 4.1 we talk about model limitations, in section 4.2 we will show the results of spaced out incremental tests conducted during the training of maps, including the spaced out incremental introduction of categories during the phase of learning and the consequences of the extension of the SOM organization phase. In section 4.3, we will present the theoretical assumptions investigated previously by Bernd Frizke in [5] for the expansion of SOM with topology organization similar to that used

in our model and the results he has achieved. In section 4.4 we show an adaptation of the technique proposed by Fritzke for the hexagonal topology of our model as an alternative technique to the dynamic expansion of maps, showing how this leads to a decrease of the quantization error and, consequently, exceeding the limit of the number of words learned by model previously proposed by Mayor & Plunkett in [21]. In section 4.5, we will present the results obtained by training using static maps, compared with the same training conducted with maps that can evolve with the previously proposed technique, demonstrating the crucial role played by the size and topology of the maps in the model. In the last section, as usual, a brief conclusive summary of what was said before.

4.1 LIMITATIONS OF THE MODEL

The authors identify many limitations in the model: we list some of them, then we identify which of these limits have been the subject of interest in this work.

4.1.1 *Visual input pattern and formation of the categories*

For simplicity, the authors have created pattern consisting of random configurations of black dots. Although these random dot figures are intended to capture the sparse, natural clustering of many object categories observed in the real world [23], it is not the case that all categories honour these sparse clustering criteria [21]. Obvious exceptions include tools, furniture, games and other artifactual inventions: furthermore, no analysis about scale, rotation and position of visual patterns (as done by P. Paplinski using Radon transform in [26]), was taken into consideration, not to divert too much from the main objective.

4.1.2 *Auditory input pattern*

The model ignores the role of time and the incremental nature of word-form processing, nor does it compute any abstract representations of phonetic or phonemic constituents of the speech signal [21]. Another

aspect of which we have not cured regards the generation of sound patterns, for which a generation technique was used like that used for visual pattern.

4.1.3 *The inability to learn new words*

The inability to learn new words after the visual and auditory maps have stabilized also compromises the plausibility of the model. This is the limit of which we have addressed here: in other words, the model is not capable of learning new words if they are presented after the stage of categorization required in the maps. To overcome this limitation, the authors suggest that a potential solution is to employ hierarchies (or heterarchies) of maps in both the visual and auditory pathways of the model, mimicking aspects of the organization of visual and auditory cortex. On the other hand, we have acted differently from what is suggested, making three distinct fundamental steps: the first concerns the **dynamic change of the training set** during the training map, designed to simulate the incremental introduction of exemplars from new categories of maps during training, to study the presentation of new "words" in the model and then exceed learning limit mentioned before; the second, concerns the **introduction of Hebbian training during SOM training**, to make less rough the separation between the organization phase (involving SOM) and Hebbian training (involving both SOM and synapses between them) present in the original model, in order to make the training for association more psychologically plausible; the third consist in **introduction of Growing SOMs** to simulate visual and auditory cortex. In order, the incremental introduction of the new categories during the standard training of the SOM does not lead to good results. Some tests conducted with the spaced out introduction of categories, in which these are presented to the maps *during* training, have brought to light an important limitation in the way they are recognized: iterating the process of insertion and maintaining fixed the size of the maps, latter become saturated and begin to *overwrite* the previous knowledge about the old categories learned to store new categories presented later. Conceivably, changing the training set during SOM organization phase is a very delicate step, that can be accomplished in dif-

ferent ways: we have identified a number of dynamic change strategies of exemplars-set, which will be discussed in depth in section 4.2. Regarding the Hebbian training introduced incrementally during the SOM training, we recorded excellent results regards: the formation of Hebb synapses introduced early and the maintenance of the same do not affect in any way the formation of synapses introduced by subsequent trainings, when the maps have achieved a better degree of categorization. We'll talk more in detail about this aspect in subsection 4.2.4. These two initial steps have confirmed the need to have the basic neural structures able to evolve. This coincides with the third step, the introduction of Growing SOM, which we will discuss in the second part of the chapter, precisely in section 4.3.

SUMMARY The result of experiments conducted presented in the following pages retrace the steps made. To summarize, in order we will touch on three key points:

- Step 1 - Presentation of an analysis of the behavior of SOM following the spaced out introduction of the categories during the training phase;
- Step 2 - Some considerations about the Hebbian training introduced incrementally and, subsequently, an analysis of the performance achieved (in terms of taxonomic response) from the whole model in the presence of both spaced out introduction of the categories and incremental Hebbian training;
- Step 3 - The last and more important feature introduced: the creation of two Growing SOMs, able to adapt to the described forms of training, created for overthrowing the limits on the number of learned categories. Some preliminary tests lead us to believe that the model is able to automatically adapt to the training set which is subjected, leading to strong performance advantages in the presence of the form of incremental training described above, most likely the type of "training" which is subject a child in the first twenty-four months of life;

4.2 STRESS-TESTS AKA *how-to-put-in-trouble* A KOHONEN MAP

In the next pages, we talk about Step 1 of three mentioned, or SOM training with spaced out introduction of categories.

4.2.1 *An alternative way to see things for the first time*

Imagine to impersonate a child a few months old during a circumstance of real learning: how could you realize this situation? There is no accurate way to answer this question, because there is no defined way to *wrap a learning event* and *isolate it* from all the other events that permeate the life of a child. We think that the importance of a simulation lies in shaping a likely situation, the most similar to what may occur in the real world. Regardless of how the model organizes itself, the same concepts of training and testing should be rethought: in a real context, there is no separate learning phase, after which we are able to use forever the concepts learned. In this sense, we can learn the meaning of a word and forget it, and then learn it again later, maybe not even remember having already *saved* in the past the word-meaning association. This is not to say that we tend to forget everything we learned in the past; however, it is reasonable to think that we are continually subject to recurring stimuli and, occasionally, new ones. Based on this idea, we tried to simulate a form of *incremental learning events* more likely the ones to which we are subject in everyday life: to achieve this, we have worked about how new objects are involved during the training phase. Technically, we can not break the phases of training and testing in our models. We can introduce some phases of testing during training, to study the trend: however, this is not the idea that realizes more precisely the scenario described above. To simulate the occurrence of events that lead us to perceive new stimuli, we must *vary* the set on which training takes place: we decide to investigate *when* certain objects are seen for the *first time*. The first time in which we see new objects are not temporally limited to the first years of life, or other life-limited periods: these first times are distributed in no particular order, with different distribution, during all life. The new stimuli, occasionally with old stimuli already categorized, after being involved in the ongoing

process of categorization, will give rise to new categories, or specialization / generalization of the previous ones[1]. As already said, changing the set of training during the SOM training does not lead to good results. It's clear that to full simulate the scenario where new stimuli are inserted randomly in the learning phase, it is necessary to have a structure that is able to adapt (in complexity) to the problem. What we want to emphasize is that, regardless of the mental model (theoretical or physical) through which we see the ability to categorize, although limited to a specific type of stimulus (i.e. visual or auditory), the cognitive structure at the base will necessarily be forever subject to new stimuli, never seen, and must be able to adapt and possibly evolve itself, creating new *computational units* (in this scenario, enrolling more neurons), to categorize and store new stimuli.

4.2.2 *A dynamic training set*

MANY WAYS TO INTRODUCE NEW CATEGORIES There are many ways to vary the training set during learning. The initial training was set up using standard 12 distortions for each categories, chosen randomly among the 24 generated from each prototype. In total, to train maps to recognize 100 categories, we have a training set of 1,200 exemplars. The variants analyzed include the spaced out incremental, not incremental and semi-incremental introduction of categories.

- Incremental way: learning with spaced out incremental introduction expected to commence training (suppose a total of 1000 training epochs) making use of a training set of 12 exemplars per 10 categories. Each 100 epochs of training, the training set is enhanced by introducing an additional 120 exemplars from 10 other categories. So, for the first 100 times of training, the map is trained using 120 exemplars, from epochs between 100 to 200 are used 240 exemplars

1 The concepts of specialization and generalization are very thin and we will talk about these in more detail in the final part of the chapter. During this work, we have not taken care of this aspect but we assumed some future development scenarios: we will present some alternatives explored to investigate these two aspects of the capacity of categorization in more detail in section 4.

(120 new +120 old)... up to the epochs slot $900-1000$ in which all exemplars are used, precisely in 1080 already seen at least once, and 120 from the latest 10 categories introduced.

- Not incremental way: learning with spaced out not incremental introduction expected to commence training (suppose a total of 1000 training epochs) making use of a training set of 12 exemplars per 10 categories. Each 100 epochs of training, the training set is replaced by introducing 120 new exemplars from 10 other new categories. So, for the first 100 times of training, the map is trained using 120 exemplars, from epochs between 100 to 200 are used 120 exemplars (120 new)... up to the epochs slot $900-1000$ in which 120 exemplars are used, from the latest 10 categories introduced.

- Semi-incremental way: learning with spaced out semi-incremental introduction expected to commence training (suppose a total of 1000 training epochs) making use of a training set of 12 exemplars per 10 categories. Each 100 epochs of training, the training set is replaced by introducing an additional 120 exemplars from 10 other new categories but keeping x (for example, 4) exemplars for each categories seen in the previous epochs slot. So, for the first 100 times of training, the map is trained using 120 exemplars, from epochs between 100 to 200 are used 160 new exemplars (120 new +40 old, 4 for each category)... up to the epochs slot $900-1000$ in which 480 exemplars are used, precisely in 360 already seen at least once in the previous epochs from 90 categories, and 120 from the latest 10 categories introduced in the slot $900-1000$.

The categories may of course be introduced in a spaced out manner by making use of a finer grain: for example, 2 new categories could be introduced every 20 epochs of training[2]. However, the way in which the training takes place depends not only on how the training set varies: as we mentioned in the end of chapter 2, there are two distinct phases during the training of a SOM. In the first one, basically half of epochs used during the entire phase of training, the update of the weights is more intense: this is due to the fact that the learning rate and the neighborhood radius

2 $1000/20 = 50$ slots, $100/50 = 2$ new categories for slots

have higher values, consequently, the number of units involved and the changes made to individual values in the weight vectors are greater than during the second phase. The behavior of the SOM after training conducted with spaced out introduction of new categories, and consequently the outcomes of Hebbian training, are different depending on the changes that occur in the training set used. In the next three paragraph, we will show and discuss the progress of the quantization error obtained respectively from spaced out incremental, not incremental and semi-incremental categories introduction during standard SOM training.

INCREMENTAL WAY In Table 6 is shown how the training proceeds with spaced out incremental introduction of categories. As we can see the quantization error tends to increase: this is natural. The training set keeps growing, and every 100 epochs 120 exemplars from 10 new categories are introduced. The map is initially able to adapt so as to categorize adequately also the new categories, but performance drops after presentation of new categories in first epochs slots.

Epochs slots	Quantization Error
0-100	≈ 12
100-200	≈ 14
200-300	≈ 16
300-400	≈ 18
400-500	≈ 19
500-600	≈ 20
600-700	≈ 21
700-800	≈ 22
800-900	≈ 23
900-1000	≈ 23

Table 6: Quantization errors during standard SOM training, with spaced out incremental categories introduction every 100 epochs, with 12 exemplars for each categories

As we will see later, this behavior will impact negatively on the performance achieved by the model as a result of the phase of Hebbian learning. The performance of the map affect the Hebbian training especially in function of uncertainty, namely the quantization error, expressed by the map on single BMU which is activated during the presentation of a word-object couple: the map is more confused, more connections created will be weak, and it will blend more easily when they are used to induce the activation on the corresponding map, during both production and comprehension tests.

NOT INCREMENTAL WAY If we look at the column of Table 7, we see the trend of the quantization error with a spaced out not incremental introduction of categories. The behavior is the same than before, because the

Epochs slots	Quantization Error
0-100	≈ 12
100-200	≈ 13
200-300	≈ 14
300-400	≈ 14
400-500	≈ 14
500-600	≈ 14
600-700	≈ 15
700-800	≈ 15
800-900	≈ 16
900-1000	≈ 31

Table 7: Quantization errors during standard SOM training, with spaced out incremental categories introduction every 100 epochs, with 12 exemplars for each categories

training set is literally **replaced** each time. However, if we analyzed the behavior of the whole model in action (we are talking about production testing after the introduction of the Hebbian training), we would notice that the performance on the learned categories are opposite: the mistake, which in the previous case was accumulated more easily on the latest cat-

egories, in this case tends to increase more and more on the categories learned in the first epochs slots, because the boundaries of them become more and more blurred after the introduction of new categories.

SEMI-INCREMENTAL WAY Quantization error trend spaced out semi incremental categories introduction is shown in Table 8. The overall behavior is already known: the semi-incremental way allows the maps not to overwrite at all the inherent knowledge about old categories, leaving spaces to new categories to be stored. The quantization error of the maps

Epochs slots	Quantization Error
0-100	≈ 12
100-200	≈ 13
200-300	≈ 15
300-400	≈ 17
400-500	≈ 18
500-600	≈ 20
600-700	≈ 21
700-800	≈ 22
800-900	≈ 23
900-1000	≈ 24

Table 8: Quantization errors during standard SOM training, with spaced out incremental categories introduction every 100 epochs, with 12 exemplars for each categories

is approximately similar to that committed with the not-incremental introduction, however, the tests conducted after the Hebbian learning demonstrate a general increase in learning performance categories, with results that are positioned more or less in half of those obtained along the spaced out incremental introduction (additive) and spaced out not incremental (with whole replacement of the training set). Based on this result, we decided to run a test by increasing the phase of organization: the idea is to force the map to store in a more incisive way new categories, at the cost

of losing the ability to categorize some of the exemplars seen in previous epochs.

4.2.3 *A test with expanded organization phase*

We decided to use the same training parameters calculated using an interval of 1000 epochs training with values similar to those used before, by restricting the use of the first 100[3] values of learning rate and neighborhood radius for each slot of training epochs: this allows us to reset the descent of the learning rate and neighborhood radius, conducting a parametrically identical training for each time slot of 100 epochs. The first test that we conducted was done using non-incremental spaced out introduction of the categories. This combination, consists in training the same map with the same training parameters using 120 exemplars from 10 categories, for 10 times, each time for 100 epochs and each time using the same training parameters (high learning rate / neighborhood radius) and replacing the 120 exemplars used in every previous 100 epochs with the new exemplars from 10 new categories introduced. The results are very poor: using the training parameters identical to those used in the first 100 epochs take the map to forget the old categories, which are *simultaneously* seen only with the categories belonging to the same set of categories introduced together. The test conducted later similar to the one described has involved the semi-incremental techniques of categories introduction. The idea is to allow the map to learn new categories with a stronger form of learning, while continuing to see some of the distortions previously learned. The results are best lightly, but the restriction to 100 training epochs for each bunch of categories, despite the injection of some exemplars (i.e. 4) for past categories, is not an optimal training condition.

Random mode Before passing to the expansion of the maps, a last test was conducted by placing a random number of exemplars for a random number of categories, for 10 times, for 100 epochs training for each

3 The phase of organization actually extends at least up to the epoch 400, however, we conducted some tests using 500 training epochs, but the organization is too strong and the map forget permanently thing has no way of reviewing in next training.

time, incrementally[4]. The result leads the model to have stable (poor) performance.

FOLLOWING In the next pages, we talk about Step 2 of three mentioned: the behavior of incremental Hebbian training during SOM training. This step is critical and we need to talk about it separately because the tests conducted later will use the spaced out semi-incremental introduction of categories and Hebbian training introduced incrementally to simulate the introduction of new words. The last ingredient will be the introduction of Growing-SOM, which will make this kind of training, more similar to that of a real context, effective.

4.2.4 *Incremental Hebbian training*

As mentioned at the beginning of the chapter, the second step taken to overcome the limits of learning new words is related to the introduction Hebbian learning during the SOMs training. To understand the learning behavior of incremental learning by association, forget for a moment the talks concerning the spaced out introduction of categories mentioned in the previous pages. Suppose we have available the two SOM and perform a standard training, similar to what was done in the last demo discussed in the previous chapter (subsection 3.5.2). We define the Hebbian training *incremental* when it is introduced every x epochs during the training of the two SOMs, keeping the previously created synaptic connections. This type of training was conducted, as already mentioned, to overcome the strong separation (not present in a real learning context) between the two phases (SOM training and Hebbian training), but also and above all to verify another important aspect of the model: the ability to not be influenced by synaptic associations prematurely created, between categories still not well defined, during subsequent training, in which the associations are established between developed (and tended to be much better) defined

4 So as to allow the map to see at least once at least one example for each category.

categories in both maps. To illustrate this phenomenon, we conducted a test[5].

IN DETAIL In particular, every 100 epochs of training of the two maps we introduce Hebbian training. For 10 consecutive times, at a distance of 100 training epochs, the *same Hebbian connections matrix* is used to perform the training, and subsequently is used to accomplish the same production test conducted in the subsection 3.5.2. The mathematical model justifies the formation of stronger connections after a greater number of map-training epochs: the activation of neurons and the intensity with which the synaptic connections present between them are reinforced are linked, in reverse, to the quantization error supplied from the map for the BMU selected. Quantization errors, with time, decreases: in accordance with what said above, the maps respond with less *uncertainty* as they arrive at the convergence phase, and this aspect is reflected in the establishment of stronger synaptic links. This explains the reason of the success of incremental Hebbian learning: associative learning, if introduced prematurely, has bad performance but synapses created in the early stages are dislodged from synapses stronger in subsequent stages of training. Practically, the old synapses *lose appeal* in the model, and although remaining active, the connections prematurely established between neurons not-yet-fully-trained will be no longer selected in the subsequent production testing, because there are other stronger synaptic connections (the youngest ones). The results, for each 100 epochs, of Hebbian training and production testing introduced without erasing the previously learned synapses are shown in the Table 9.

5 This experiment was conducted in a similar manner also by Mayor & Plunkett in [21], but not incrementally.

Stats description	Visual Q. Error	Auditory Q. Error	Absolute count	% of categories learned[6]
Epoch 100 k	26.4	11	435/1200	36.5%
Epoch 200 k	25.6	10.7	615/1200	51.5%
Epoch 300 k	24.9	10.5	660/1200	55%
Epoch 400 k	24.5	10.2	708/1200	59%
Epoch 500 k	23.9	10.1	768/1200	64%
Epoch 600 k	23.4	9.9	815/1200	67.9%
Epoch 700 k	23	9.9	883/1200	73%
Epoch 800 k	22.6	9.8	921/1200	76.8%
Epoch 900 k	22.1	9.7	948/1200	79%
Epoch 1000 k	21.7	9.6	983/1200	81.9%

Table 9: Result of production test every 100 epochs training with 12 exemplars for each categories, and one-shot labelling event (hebb training) for each categories, tested with others 12 exemplars for each categories (different from the ones used for maps training), using the same Hebb Connection matrix in each training/testing.

TRIAL RESULTS In Appendix, are show incremental Hebbian testing grouped result for categories in the following configuration: spaced out incremental, not-incremental and semi-incremental introduction, both with and without expanded organization phase during SOM training.

4.2.5 *Problems with maps: the idea of enrolling new neurons*

In the preceding pages, we have introduced two new learning techniques: the introduction of spaced out semi-incremental categories and the incremental Hebbian training during SOMs training. Furthermore, we saw that expand the phase of organization at the expense of that of convergence does not lead to good results. One of the known problems of self-organizing maps is linked to the fact that, by increasing the number of categories that these must be able to identify, the available space for each of them, in terms of number of neurons on the map, decreases. We pay attention to one fact: the problem does not consist in the number of exemplars presented to the map. In fact, given a map with an arbitrarily small number of neurons, this is able to best align itself to an arbitrarily large number of exemplars: the problem consists in the result that we

want to obtain when we subsequently present an input pattern. If the patterns are distinguished between two possible categories, then it takes just a few neurons (potentially, exactly 2) to make the map able to distinguish them. If the patterns presented in the input must be distinguishable in hundreds of different categories, then it is natural to think that hundreds of neurons must be trained. Introducing also the need for natural noise recognition, namely, in our examples, the ability to distinguish noisy exemplars as belonging to categories from which *deviate* in part, is still more clearly the reason for which the number of available neurons is critical in whole training process. So we decided to increase the size of the map and see how this aspect influence on the ability to categorize new categories, tested again by introducing new exemplars in a semi-incremental way and incremental Hebbian training every 100 epochs of SOM training. We will discuss about this enlarged fixed configuration of maps later, in section 4.5. We anticipate that the results are very good but we don't like the idea of solving the learning problem by incorporating a greater number of neurons at the beginning: for several reasons, first of all because this solution is not a real solution. For an arbitrarily large number of categories, we have reason to believe that the problem of maps' size reoccur easily. A question arises spontaneously: what would happen, if the map was able to enlist new neurons when necessary, without the need to determine in advance how many of these serve to maintain a particular performance? In the next section, we will introduce the dynamic expansion of the map, namely the introduction of Growing SOM in the model.

4.3 THE DYNAMIC EXPANSION OF THE MAPS

4.3.1 *A brief introduction*

The last and most important step has finally found its justification: the idea is to transform the standard sequential SOM training by introducing the ability to evolve according to certain parameters, enlarging neurons grid while preserving the invariants of topological structure and allowing to use the same Hebbian connections developed with previous map topology. The whole process will be explained in detail in the following

pages: before introducing the dynamic expansion of the map, let's talk about some solutions proposed in literature about evolving SOM. Then, we will introduce of the original model proposed by Bernd Fritzke, from which we took great inspiration for the creation of the dynamic maps expansion.

4.3.2 *State of the art*

One of the first jobs that we find in the literature about learning patterns of language is the work of Plunkett, Sinha, Møller, and Strandsby [27]: in this work and in Associative approaches to lexical development [24], it is presented a auto-associator. This model combines label to image and is based on previous models like that of Knapp & Anderson [10] and the extension of Chauvin [4]. Input patterns created are presented according to Posner's technique [29, 31]. An interesting aspect of this work is that before the submission to the network, all images pass through a *retina* that compresses the representation of the input pattern from 2100 px to 171 retinal cell, contemplating that form of generalization reached by SOM in our model. Then the retina stores a concept of proximity between generalized patterns. One of the shortcomings of this model is related to the use of auditory patterns consist of 32 binary vectors in which a only single bit is active: they are completely orthogonal and there is a categorical structure between the label (the association with the visual categories is arbitrary as in our model). The specific task of the network is to reproduce distinct representations of the images presented to the retina and of the label: then, the network is trained with a combination of self-learning phases. The middle level of hidden units creates a composite representation: following the presentation of an image, is possible to get the label (production test) and viceversa (comprehension test). The hidden units, immediately after the retina and the level that identifies the label, are used to achieve fine-tuning: for example, from the side of the visual recognition, these units are used to make fine-tuning of the clustering work done by the retina level. A conscious critique to this model, and in general to the different types of models available in the literature is moved by Regier in 2005 ([32]), in an article in which he presents LEX.

In his work, it is reviewed three types of models: subsymbolic, competition and inference models. LEX is the first work that manages to combine together the features of subsymbolic and competition models: the model is presented as a mix between the model of Plunkett & Co (1992), mentioned before, and ALCOVE by Kruschke, [12]. Reiger categorizes the first one as a subsymbolic model and stresses that «this class (subsymbolic) of models fails to capture children's ability to remember a newly learned word in the face of up to a month's Subsequent exposure to other words» - Markson & Bloom, 1997 [20]. This limit will still be present in the model presented by Mayor & Plunkett in 2010, [21], but as we will shown, the model described in this work, an evolution of the model they proposed, overcome this limitation. A strong criticism from Reiger is linked to the fact that lot of models in literature suffer from a severe lack: a phonological representation and, consequently, the inability to take into account the change of the ruling sensitivity. This criticism is also moved by Mayor & Plunkett to their model. In later models this lack is filled and it increasingly recognizes the importance. In our model, this type of failure is not really present: this does not mean that we have a solid phonological representation, but we have a growing SOM whose sensitivity to the pronunciation is improving because the words are categorized always better during subsequent phase of training.

An interesting aspect that concerns the lexical evolution and that will be featured in all subsequent models, is introduced by the CALLED model (Merriman's - [22]) and consists in selective attention to certain input dimensions. This type of selective attention can also be realized with SOM applying a mask from different importance to each of the weight vectors that characterize neurons. A model that tries to explain language learning in adults is ALCOVE: this implements this form of selective attention through attentional weights. We will return to this point in the conclusions.

ALCOVE, a model presetend by Kruschke in [12], is a three-level model: his strenght is that *combines* in one model two different approaches: exemplars theory and prototypes theory. According to Nosofsky ([25]), the author of the GCM model that underlies ALCOVE, many network models flatten out to be prototypes models because the stimuli are categorized

according to their proximity to "centroid" of categories. Indeed, even the SOM model of the Mayor & Plunkett works in that way. According to Kruschke, the goal of ALCOVE is not to discover new representations in the hidden layers after many trainings, but to drive trainings discovering how much different input dimensions matters more. The model consists in an input layer, an intermediate hidden layer in which they are stored exemplars, and an output level. The approach is connectionist but there is also a form of interaction between exemplars in the middle level. It is a good model because it presents the attentional weights mentioned previously in input nodes, and these weigths are used as multipliers when calculating distances between stimuli and nodes belonging to hidden level, to give more or less importance to certain input dimensions (features). One of the shortcomings of this model is related to the number of categories (number of output nodes): this is defined upstream. The model that we present in this work does not have this drawback: if there is a sufficient number of neurons, a SOM is able to categorize an arbitrarily large number of categories. Another fundamental flaw of these models, is so far linked to the correction of errors: the connections of the ALCOVE and also the previous model is always error-driven, then linked to a form of supervised learning. The Hebbian learning, however, does not provide error correction is supervised in the sense that stimuli are taken in pairs (defined not by chance, or group are correct) but the supervision is limited to this aspect. Despite this flaw, ALCOVE present no catastrophic interference, because the *storage* of exemplars in the hidden level prevents it. «None of the models described above exhibited taxonomic responding and fast mapping since they Involved training networks on the repeated and multiple object-label associations in order to extract the statistical structure of the label-category relationships» - Mayor & Plunkett, [21]. Another criticism that is easily raised is that everything that makes use of back propagation or any type of algorithm that is based on gradient descent needs a constant supervision and multiple exposures to stimuli. The only way to cope with this necessity and use a single labeling event to obtain a generalization of the word-object association passes through the use of the SOM.

SOM BASED MODEL One of the first models based on SOM is presented by Schyns in his job in [34]: the model has a structure similar to the one used by Mayor & Plunkett in [21]. However, although the idea is innovative, the model is tested on a limited number of categories, but the most interesting aspect concerns the theoretical limit highlighted by Schyns about the very idea of *concept*. According to Schyns, humans form concepts to represent things that, in reality, have no a priori similarity between them: we are fully agree with this statement. If we take the categories of things like "be healthy", then "follow a good diet", "do some exercise," and "enough sleep" are activities that have few properties in common - Barsalou, [2]. How can they be grouped? Murphy and Medin say that people can form categories according to a "theory" they have about the objects in these categories. In practice, there is some sort of investigation in the categorization process, which provides an analysis of what is already known of objects in that category. We share the idea of Schyns according to which these theories are talking about Medin and Murphy could highlight the information that "is" and "is not" important, to take into account different situations thus maintaining the conceptual coherence of the category: conceptually, the process realized by the presence of these theories could be implemented by a sort of features selection. So according to Schyns humans do feature selection, and this aspect is also heavily discussed by Reiger.

The most interesting models are those of Li and colleagues [14, 15], Devlex and Devlex II. The first is a model that seeks to solve the problem of growing words number with Growing-SOM. In their work, Li and colleagues dismiss the proposal growth assumptions from Fritzke ([**?**]) and others. Their technique to cope with the problem of catastrophic interference after new words presentation is based on the combined use of static SOM and ART2 model (Carpenter & Grossberg, 1987 [3]): during their growth process, there is a transition from SOM model to the other, ART2, and the idea is to enable silent nodes, previously disabled, that can be enrolled when needed. One of the defects is related to the operation of ART2 mechanism: in practice, when there is a change from one model to the other, SOM concept is set aside, topology is ignored, and so they must use another method (Distance Ratio Preservation) to find

the right position of the new node in the map. This process undermines the plausibility of the model, in a more consistent way than what was achieved in the proposed model. You could move a critical compared to the enlistment process of new neurons: enable silent neurons is definitely more neurologically plausible. However, we believe that recruiting new neurons, or having a set of silent units are two different ways to interpret the same problem.

SUSTAIN One of the most suitable model to categorize is SUSTAIN, presented by Love & Co. in [16]. SUSTAIN (Supervised and Unsupervised STratified Adaptive Incremental Network) is a model of how humans learn categories from examples. SUSTAIN initially assumes a simple category structure. If simple solutions prove inadequate and SUSTAIN is confronted with a surprising event (e.g., it is told that a bat is a mammal instead of a bird), SUSTAIN recruits an additional cluster to represent the surprising event. Clusters compete to respond to input patterns and in turn inhibit one another. When many clusters are strongly activated, the output of the winning cluster is less: psychologically, this signifies that competing alternatives reduce confidence in a choice. The SUSTAIN model, which aims to model a unique response to categorization problems, including inference problems, category constructions, etc., is packed with parameters: in [16] are listed many of these, including some about attentional focus, cluster competition, decision consistency, learning rate, category focus, and others. Moreover, SUSTAIN's cluster recruitment mechanism creates a new cluster when the current item is not sufficiently similar to any existing cluster. This threshold is captured by another parameter. In order to set all these values, SUSTAIN's parameters are adjusted to minimize the sum of squared error among data means and the means calculated by averaging over thousands of SUSTAIN simulations. Although, according to its authors, «SUSTAIN's behavior is not extremely sensitive to the particular values of the parameters», we have reason to believe that this type of operation is too complex, especially for the type of task we have to solve: it is universally accepted that the cognitive skills are subject to development in early childhood. This model seems to be moving in the direction of justifying a huge number of cog-

nitive processes, far more articulate than those that guide the learning of words in early childhood. On the other hand, SOMs are very good for reproducing plausible cognitive processes, by very reason of their simplicity: the model, for training, not only defines a vector space on which to run the accounts. In contrast to the SUSTAIN model, a SOM is an topologically organized model, and this feature is a double-edged sword: on the one hand, this organization fits easily in contrast to the high dimensionality input and makes it difficult to always represent in a correct way the proximity between the categories in topological terms, especially when the neurons are arranged on grids with a low number of dimensions. On the other hand, the presence of a topological order, and the influence of this during the training phase binds in a deeper way to the nature of the human brain, in which the neurons are actually connected to each other by synapses. The interaction between neurons takes place in a more coherent way with the type of process which the model tries to simulate. This led us to think of a natural evolution of the yield originally proposed by Kohonen maps: the growing self-organizing maps. A growing self-organizing map (GSOM) is a growing variant of the popular self-organizing map (SOM). The GSOM was developed to address the issue of identifying a suitable map size in the SOM: exactly the kind of goal that we defined earlier. Many different models have been created to achieve this sort of incremental SOM, able to evolve. The literature about proposed arbitrarily complex models, such as those described in [26] and [39], however, our focus has shifted quickly in the direction of a linear and simple solution that did not involve distortions of the original model, taking into account any previous considerations leveraging standard SOM. With this goal in mind, we have selected the work of Bernd Fritzke, who exposes his solution to the problem of automatic expansion without detaching too much from the prerequisites of our model. In the next section, we will deepen the model presented by him in [5], explaining in more detail the growth process: then we will explain how we have modified this technique to overcome the limit of learning new words, with the introduction of our version of Growing SOM.

4.3.3 *An old idea by Bernd Fritzke*

In *Growing Grid - A self organizing networks with constant adaptation neighborhood and strength* - [5], Bernd Fritzke present a novel self-organizing networks which is generated by a growth process. We chose this model because the objective pursued by Fritzke was in line with our needs: create a neural structure from a rectangular grid that it was able to increase its size during the self-organizing process. The technique proposed by Fritzke allowed us to attack the problem of the expansion of the map as simply as possible, responding specifically to two fundamental questions: *when* we proceed with the expansion, and, fixed the moment, *how* we could proceed? We recall that our lattice, for comparison with the needs of Mayor & Plunkett model and to make treasure of the studies formulate by them about that particular configuration, is hexagonal, in a two-dimensional grid. A first major issue is to preserve the topological structure: is not possible, in fact, add some neurons to the map, without relying on the topological nature of the space defined. To put it through an image, if we look at the map shown on the left in Figure 25, this can not be expanded as shown in the immediately right image: if necessary, you can add an entire column or row (third and fourth figures). This constraint involves a number of considerations which we will return later. Suppose, to continue the argument, to accept the idea of adding a row or a column (or possibly both) to the map. A second, trivial question, is: where we add new neurons? At the boards, in a random position, or in the middle? Knowing the dual nature of the multidimensional space defined by the weight vectors of the maps, we could imagine that perhaps there is an optimum topological position (or more than one) in which to place new neurons. The work of Fritzke, in these terms, was enlightening.

FRITZKE MODEL: EXPLANATION Fritzke uses a rectangular map and associates to each neuron a vector of weights and a *resource variable*, a simple counter initialized to zero. The resource variables are used to gather statistical information to decide where to insert new rows or columns of units in the network. Subsequently, the training of the map proceeds as

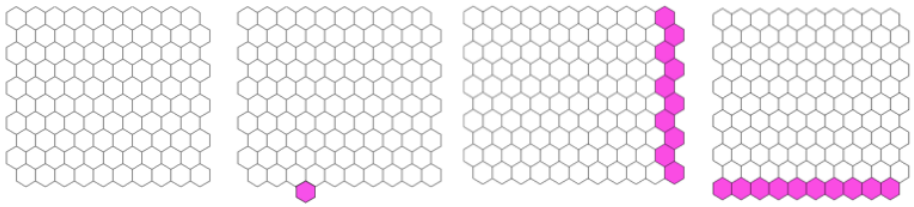

Figure 25: A first approach to the expansion of a hexagonal SOM.

defined by the Kohonen. In addition, at each adaption step the resource variable of BMU is incremented:

$$\tau_s = \tau_s + 1; \tag{13}$$

and, therefore, these values show how often a unit has been BMU. For a network of size $k \cdot m$ he performs $k \cdot m \cdot \lambda_g$ adaption steps before inserting new units. Thus, the parameter λ_g indicates how many adaption steps on average are done per unit before new units are inserted. After $k \cdot m \cdot \lambda_g$ number of adaption steps have been performed, we determine the unit q with maximum resource value:

$$\tau_q \geqslant \tau_c \quad \forall_c \in \mathrm{Units} \tag{14}$$

This unit has been best-matching unit most often and, in order to distribute the pattern evenly over all units, it makes sense to insert new row or column in its vicinity. Since there are several possibilities how to this he have to choose a particular one. The idea of Fritzke is select the most different neuron f (in term of Euclidean distance between weight vectors) from the nearest ones to q: then, insert a column / row between the two units. The reasoning behind this choice is that f presumably indicates a direction with high variance in the underlying data. This, again, should be taken into account by increasing the resolution of the grid in the direction ([5]). In Figure 26, from left to right, are shown the BMU in green, nearest units (distance 1 in topological sense) in yellow, the most distant unit from the BMU in blue; in the right image, the column of new units in pink introduced between the BMU e the most distant. The weights of the new units will be set to the average value of the respective weights of the units next to them, in this case of the left and right columns respectively the pink one.

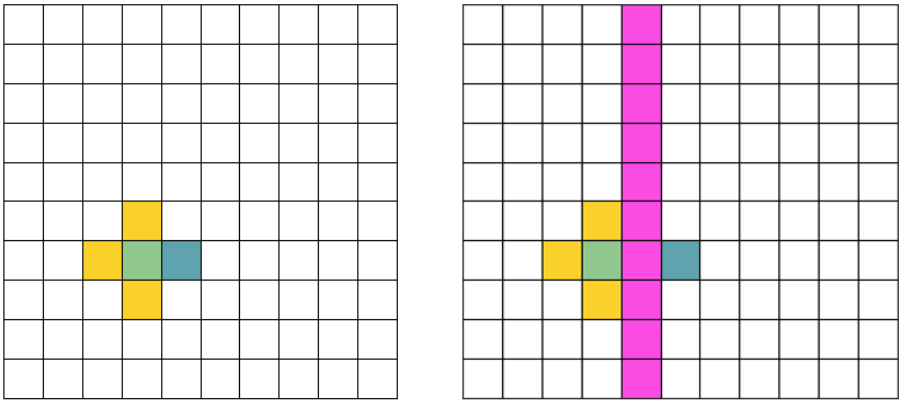

Figure 26: A first approach to the expansion of a hexagonal SOM.

STOPPING CRITERIA A normal training stop in the number of epochs defined: however, Fritzke identifies two additional stop criteria. The first is trivial: we impose a maximum number of units in the map. The second is an indirect criterion: due to the strong constraints of the topology it can not be expected that each unit receives an equal amount of input signals (i.e. a fraction $1/(m \cdot k)$). Therefore, one could «continue the growth process until the share of input signals falls below a bound for each unit in the network».

As we have seen, the idea of Fritzke is simple and powerful: for more details on the experiments conducted and results he achieved refer to the reading of [5]. Based on the work done up to this point, we have created several techniques to expand the maps, which are aimed primarily at the way new neurons are added to the map and the learning parameters used for training. We will return on the last part exposed by Fritzke about the final stage of fine-tuning in chapter 5, the final part of this work: in the next pages we present the problems encountered in the model and the changes made to the original technique he proposed. Then, we will show some comparative experiments between some selected and interesting training configurations chosen between those realized.

4.4 THE ADAPTATION OF THE TECHNIQUE PROPOSED BY B. FRITZKE

4.4.1 *The known issue*

The first issue of Fritzke technique is that this is well suited to a map with rectangular lattice: the maps used up to this point are hexagonal. How could we proceed? The first approach has been to adapt the concept of proximity to the shape of the neurons: fixed a neuron in one of the two maps, as we already shown in Figure 16, this has 6 nearest neurons[7] at distance 1 (not 4 as in Fritzke model), of which 2 to *north*, 2 to *south*, 1 to *east* and 1 to *west*, as shown in the left upper corner of Figure 27. It was decided, as a first approximation, to consider the two neighbors above and below, respectively, indicators for the addition of a line above and below the selected unit. Problems have arisen in the way the library SOM-Toolbox handles weight vectors. Fixed a map of size $m \cdot n$, the library associates to each neuron on the map an index: the indexes are assigned *per column*, as shown in central part of the figure. The weight vectors are retained in memory in a matrix, called *codebook*, of size $k \cdot j$ with $k = m \cdot n$ and j equals to the size of the weight vectors. The addition of a column within the topology, involves insertion of a series of **consecutive** rows in the matrix with the weights of vectors, as shown in the lower right matrix in figure.

7 On average, because the marginal ones has less the 6, often 3 except for the ones in the corner, which has 2 nearest units.

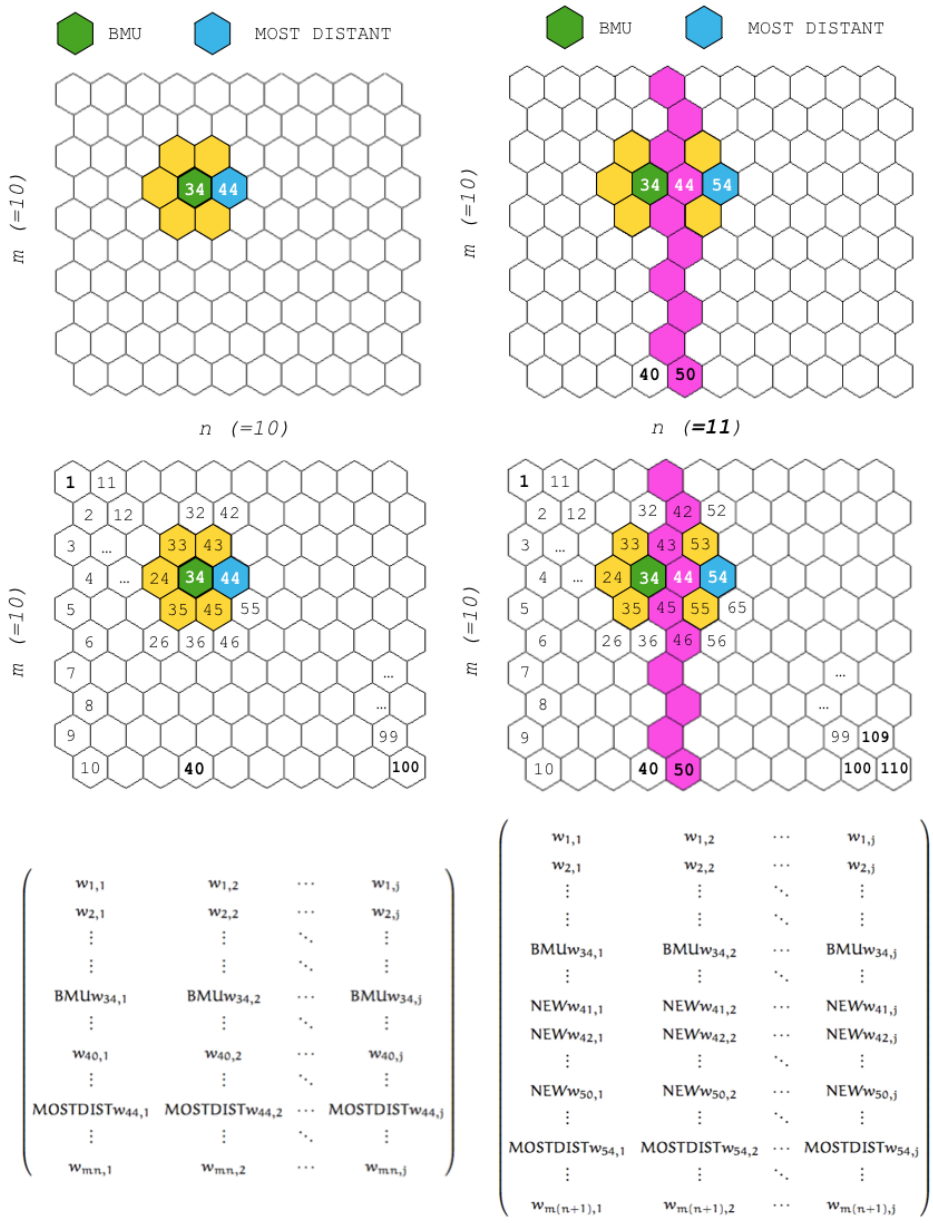

Figure 27: A first approach to the expansion of a hexagonal SOM: unit-column insertion.

The addition of a row in the topology is slightly more complicated, as it translates in the insertion of **specific rows** in a specific points in the matrix of weights, not in a *row-block* insertion as shown in lower right matrix expanded shown in Figure 27. The Figure 28 shows the two situation[8]. It

8 The left column insertion is analogous to the right column insertion.

happens that the inclusion of a row involves the insertion of a row in the matrix of weights at the index of the *first new unit inserted in the new row*, then the shift of the next x old row, then the insertion of an additional row for the *second new unit*: the operation is very delicate because it is necessary to accurately calculate the positions of the new rows, which results in a work of indexes calculation in which it is easy to make a mistake (also with risk to not recognize it). In fact, *each units in the maps change its index according to the old position and the position of the new row inserted* during process of expansion.

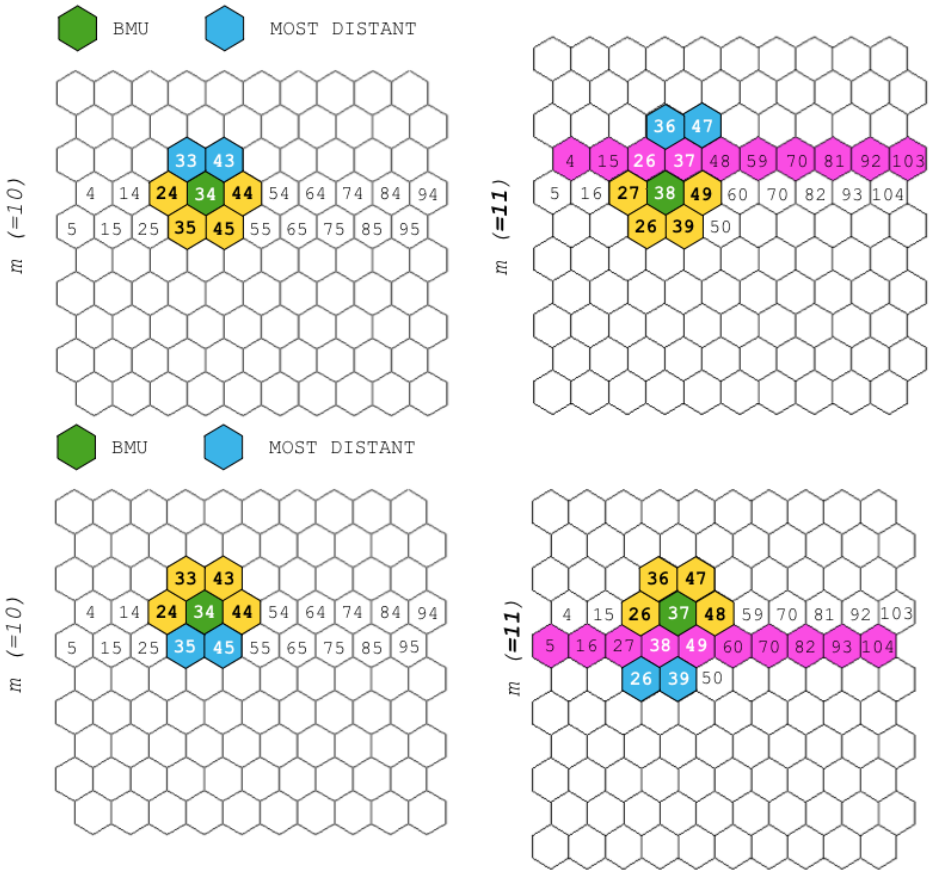

Figure 28: A first approach to the expansion of a hexagonal SOM: unit-row insertion.

A BIG PROBLEM TO BE SOLVED Although it seems that inserting rows and columns with the consequent need to maintain a consistent array of weight vectors of each neuron is a difficult problem, the most difficult

problem to deal with is not about the *codebook*, but the matrix of Hebbian connections. As we have had occasion to specify, this is defined as follows:

$$\mathrm{HebbMatrix}_{m\cdot n,k\cdot j} = \begin{pmatrix} w_{1,1} & w_{1,2} & \cdots & w_{1,k\cdot j} \\ w_{2,1} & w_{2,2} & \cdots & w_{2,k\cdot j} \\ \vdots & \vdots & \ddots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ w_{m\cdot n,1} & w_{m\cdot n,2} & \cdots & w_{m\cdot n,k\cdot j} \end{pmatrix} \tag{15}$$

where:

m is the number of lines of the visual map grid;

n is the number of columns of the visual map grid;

k is the number of rows of the auditory map grid;

j is the number of columns of the auditory map grid;

As already said, the incremental Hebbian learning provides for the old synaptic connections. If the topology of the maps remains unchanged, the array of connections connections has fixed size: simply use the same matrix of synaptic weights defined in the previous step, without *reset* the synaptic weights every time the Hebbian training is introduced, and you are done. If during the training the topology of maps changes, then the matrix of Hebbian connections must change consistently maintaining old connections and growing in dimensions to create synaptic connections between new units. So it is necessary to keep track of the changes of the map grid, in order to neatly and reflect all the topology changes also in the Hebbian connections matrix. Conceptually, introduce new neurons on the visual map means introducing new synapses between each of them and each neuron on the auditory map: preserve synapses previously created in Hebbian training and expand the matrix of Hebbian connections weights in order to make space for new synapses to develop, it's a very complex job. Let's talk about how we did it.

4.4.2 *Preserve Hebbian connections during Growing SOM training*

The matrix of Hebbian connections has a number of rows equal to the number of neurons of the visual map, and a number of columns equal to the number of neurons of the auditory map. Each expansion of the visual map, whether it takes place by adding a row or a column, results in an expansion in terms of *number of rows* in the Hebbian matrix. Similarly, each increment of neurons on the auditory map translates into an increase in the *number of columns* of the Hebbian matrix. The incremental Hebbian training is conducted basically every 100 epochs in each experiment: during the first step, the Hebbian connections matrix is created on the basis of the size of the visual and auditory Growing SOM after 100 epochs of (growing) training, so the problem does not arise. At the epoch 200, the two maps are potentially changed: not only, they could be changed more than once, and this makes it necessary to review the same changes of the topology, in the **chronological order** (i.e., in term of epochs) in which they occurred, to make the right moves in the Hebbian matrix. To make this shifts / insertions, during training we record epochs in which a change occurs, and the indexes of the BMU inserted, exploiting MATLAB *structs*[9]. These structs are then retraced and each update is executed in Hebbian matrix, in which new synapses (new rows and columns) are created between the new and old neurons, in a spontaneous synapto-genesis process similar to that simulated for the first step, however, made only to populate new rows and columns inserted. The Hebbian matrix, at the end of this complex steps, is ready to be used during the associative training phase, in the expanded maps are fully linked, with old synaptic connections all preserved. A

9 Similar to the structures in the C language, MATLAB facilities allow you to create arbitrarily complex associative arrays, to keep certain data in memory in a more elegant way: the programming style is very close to the OO one, without enjoying all the benefits that this offers but gaining in simplicity.

simplified pseudo-code of the whole process is shown in Algorithm 4.

Algorithm 4: Pseudo code of Hebbian connections matrix expansion with preservation

```
Data: HebbMatrix, Visual / Auditory maps with expansion history.
Result: Expanded Hebbian Connections matrix
previous_epoch = 1;
foreach epoch ∈ Hebbian_training_epochs_list do
    epoch_slots = [previous_epoch; epoch];
    foreach expansion on visual map occurred ∈ epoch_slots do
        visual_indexes_new_units = append(epoch,
        indexes_added);
    end
    foreach expansion on auditory map occurred ∈ epoch_slots do
        auditory_indexes_new_units = append(epoch,
        indexes_added);
    end
    foreach indexes on visual map ∈ visual_indexes_new_units do
        HebbMatrix = Algorithm 5(HebbMatrix, indexes, 'row');
    end
    foreach indexes on auditory map ∈ auditory_indexes_new_units
    do
        HebbMatrix = Algorithm 5(HebbMatrix, indexes, 'column');
    end
    previous_epoch = epoch;
end
return ExpandedHebbianMatrix;
```

THE HEBBIAN CONNECTION MATRIX - EXPANSION ALGORITHM In the algorithm, the procedures for identification of the expansion steps are not made explicit in a precise manner as how-to realize what is described in the text would result in a verbose description of certain implementation choices: despite this, for completeness we present the insertion algorithm of rows and columns, that is to say the true and proper[10] expansion func-

10 The MATLAB syntax has been simplified by introducing fictitious names of symbolic functions.

tion in Algorithm 5.

Algorithm 5: Expansion function

```
Data: HM, new_unit_indexes, mode
Result: Expanded_HM
last_copy = 1;
/* row insertion in Hebbian Matrix (visual map changes)    */
if mode ≡′ row′ then
    until = row_number(HM) + number_of(new_unit_indexes);
    foreach i = 1 to until do
        /* unit not involved in change, copy entire row     */
        if i ∉ new_unit_indexes then
            Expanded_HM = append_row(HM(last_copy, :));
            last_copy = last_copy + 1;
        /* unit involved, add row with new random weight (as
           init)                                            */
        else
            Expanded_HM =
            append_row(random_synpato_genesis);
    end
/* column insertion in Hebbian Matrix (auditory map changes)
   */
else
    until =
    column_number(HM) + number_of(new_unit_indexes);
    foreach i = 1 to until do
        /* unit not involved in change, copy entire column  */
        if i ∉ new_unit_indexes then
            Expanded_HM = append_column(HM(:, last_copy));
            last_copy = last_copy + 1;
        /* unit involved, add column with new random weight
           (as init)                                        */
        else
            Expanded_HM =
            append_column(random_synpato_genesis);
    end
return Expanded_HM;
```

The algorithm proceeds along the entire Hebbian matrix passed as a parameter, excluding from copy all the lines whose indices do not coincide with indices of units involved in change, namely the new units inserted in the map (listed in the vector new_unit_indexes). When this does not occur, it means that the index reached is the index of a new unit, so we have to insert new row (or column). In this case, a row (or column) is initialized with the same parameters used for the initial synpato-genesis that involved all the cells of the Hebbian matrix.

4.4.3 *Expansion around the neuron*

THE WINNING ALTERNATIVE An alternative way to insert new neurons is not to look for the direction of maximum variance, and insert *two* new columns and *two*[11] new rows **surrounding the selected neuron**. This technique consists in a sort of extension of the technique originally proposed by Fritzke, to achieve the expansion in a perspective of increased insensitivity to the distribution of probability with which the input pattern are scattered: assuming are uniformly distributed, and in a training situation optimal be recognized by BMU equally distributed evenly on the map, then it is natural to think that if a neuron fires as BMU more than others during training, this should be surrounded by new neurons, distributed evenly around himself. To achieve this second type of expansion, we used the expansion technique previously created: the idea is to compute the neighbors of the BMU selected to carry out the expansion, and add a row (or a column) around this by iterating on all neighboring neurons. The only aspect to which we must pay attention, is related to the fact that after a single expansion, at least one of unique indexes of the selected unit for the expansion and its neighbors units change: so, an upgrade of indexes of *involved-in-change* units must be done. In Figure 29 we see represented this second expansion technique. The image must be read from left to right, top to bottom: in the first figure, we note the neuron that has been activated more times as BMU. This has 6 neighbors: as said previously, there will be 4 expansions. In the next picture, we see

11 May be less, if the neuron is marginal, in which case we add only one row / column, for a total of 3 or even 2 additions, if the selected unit is at the corner of map grid.

the first expansion: a column is inserted to the right of the neuron. In the following, a row below. Different colors were assigned to each of the neurons at distance 1 from the BMU and used in every expansion step to highlight the moves: in the first displacement, the first two columns have been highlighted in blue to facilitate the understanding of the shift operation idea. In subsequent images, we see regarding the introduction of a line below, one column to the left box, and one line above the BMU. In the last image on the bottom right, they show the new neighbors at a distance 1 surrounding the BMU selected.

4.5 SOME RESULTS IN COMPARISON

In this last section, we will show the results obtained by incremental Hebbian training and semi-incremental insertion technique of the new categories, during a training with learning rate and neighborhood radius decreasing and repeated every 100 epochs (expanded organization phase), using, in order, two standard SOMs with fixed grid size $25 \cdot 25$, another couple with grid-size $50 \cdot 50$ and a couple of Growing SOM that expand with the technique proposed in subsection 4.4.3 up to $\approx 50 \cdot 50$ size.

4.5.1 *Static SOMs with grid size* $25 \cdot 25$

Here are the results of the incremental Hebbian training with semi-incremental introduction of categories, with static maps of grid dimension $25 \cdot 25$. In Table 10, in the first row it is shown the quantization error reached from the map at the epoch defined on the columns; in the last rows, the percentage of categories learned (with respect to those introduced defined on the rows).

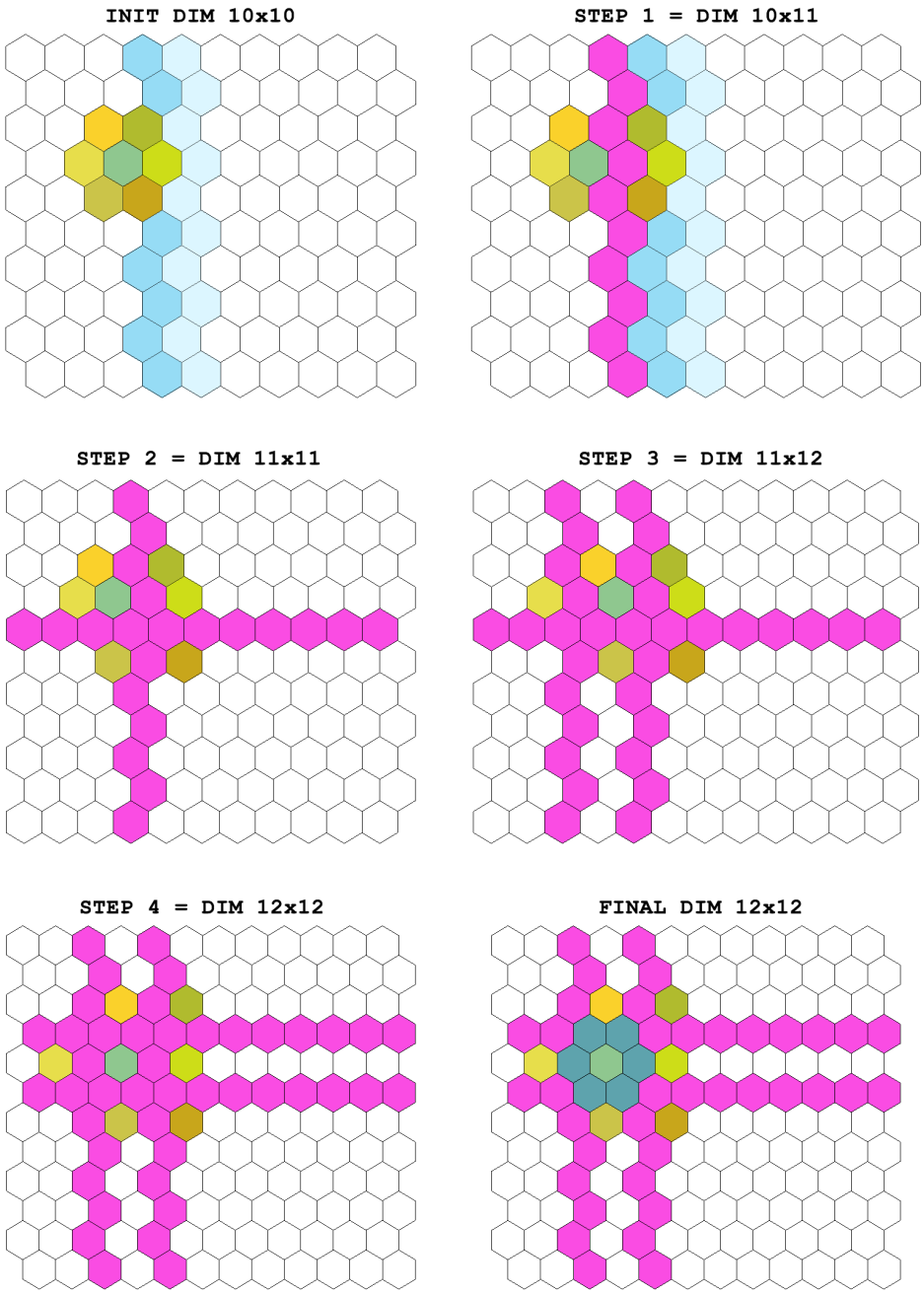

Figure 29: SOM Fritzke uniform expansion.

Categories/Epochs	100	200	300	400	500	600	700	800	900	1000
Visual Map Q.Error	11.82	12.61	15.24	16.94	18.42	20.20	21.06	22.16	23.42	25.93
Auditory Map Q.Error	5.45	5.60	6.09	6.75	7.60	8.15	8.62	9.02	9.58	10.86
Learned words	95.00%	92.92%	85.00%	84.38%	77.33%	71.11%	63.33%	57.92%	52.96%	47.83%

Table 10: Q.errors and percentage of categories learned during production test conducted after a spaced out semi-incremental introduction of categories during static maps training with grid size of $25 \cdot 25$ and a single labelling event during incremental Hebbian training.

4.5.2 *Static SOMs with grid size* $50 \cdot 50$

The two maps were expanded by 4 times (50x50 *hexa*grid). The results are obvious: the categories are learned almost completely, (final result about 88% of learned words, with $\approx$ 100% learned until the introduction of last 10 categories) because the map has a huge number of neurons available to categorize exemplars seen during training, so Hebbian training succeeds because the quantization error is minimized. However, analyzing the neurons actually used by the map to categorize patterns, it appears that many of them are not actually exploited: the *unused* neurons are neurons that have *passive role* in the complex process of categorization, acting as a *barrier* between categories boundaries, making it harder to overlap them and consequently leading to better learning, despite the incremental introduction and evolving training set. Furthermore, the problem is solved only partially, because if the introduction of categories lasts for a greater number of categories, as already mentioned, it is natural to think that the two maps will be saturated again.

Categories/Epochs	100	200	300	400	500	600	700	800	900	1000
Visual Map Q.Error	10.93	11.09	11.01	11.20	11.41	12.08	12.39	13.04	13.92	17.40
Auditory Map Q.Error	5.13	5.00	5.06	5.17	5.14	5.32	5.44	5.64	5.93	7.36
Learned words	100.00%	100.00%	100.00%	100.00%	99.83%	98.06%	99.05%	98.23%	97.04%	88.00%

Table 11: Q.errors and percentage of categories learned during production test conducted after a spaced out semi-incremental introduction of categories during maps training with grid size of $50 \cdot 50$ and a single labelling event during incremental Hebbian training.

4.5.3 *Growing SOMs from* $25 \cdot 25$ *to* $50 \cdot 50$

To test the incremental Hebbian learning performance applied on Growing SOM, we chose to start from two maps of the same size of the static first maps tested before ($25 \cdot 25$), and to set the expansion parameters in such a way as to finish the training at $\approx 50 \cdot 50$ previously tested. The results are surprising: the maps obtain performance equal to those

obtained from the instanced maps with initial larger number of neurons $50 \cdot 50$. This result makes us believe that the process of expansion of the maps, such as the expansion of Hebbian synaptic connections, is a good support to the incremental form of training introduced to cope with the problem of learning of new words. Results are shown in Table 12.

Categories/Epochs	100	200	300	400	500	600	700	800	900	1000
Visual Map Q.Error	19.79	22.23	17.91	16.14	14.74	14.80	15.26	14.99	15.72	18.77
Auditory Map Q.Error	7.67	7.39	8.32	7.36	6.56	6.63	6.58	6.26	6.37	7.70
Learned words	100.00%	92.08%	95.56%	95.42%	93.50%	95.97%	94.05%	94.69%	93.15%	84.83%

Table 12: Q.errors and percentage of categories learned during production test conducted after a spaced out semi-incremental introduction of categories during growing maps training with init grid size of $25 \cdot 25$ and final $\approx 50 \cdot 50$ and a single labelling event during incremental Hebbian training.

Categories specific results could be found in Appendix, Table 23, Table 24, Table 25, Table 26, Table 29 and Table 30.

4.5.4 *Compared results*

In Figure 30 we show the performance of three test described above in comparision histogram.

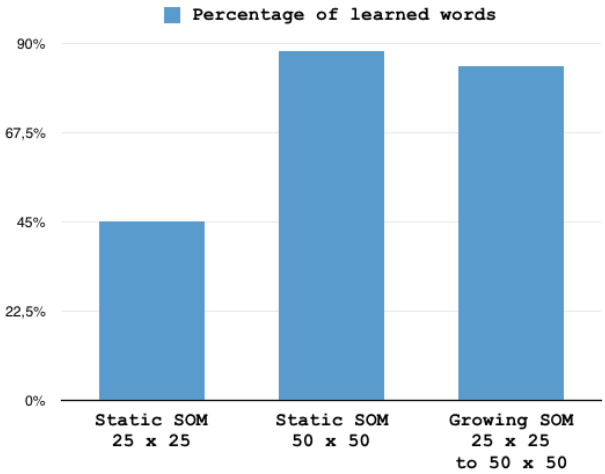

Figure 30: Compared results of static, static enlarged and growing visual auditory models.

We have reproduced many of the tests conducted by the authors of the original model to test the effectiveness of the model, we tested both forms of training, standards and incremental, on both model variants, with static maps and growing SOM. This type of tests were conducted to verify two aspects: first, the new model's ability to preserve the characteristics previously recorded during a standard method of training despite the introduction of growing SOM. Second, to compare the ability of the proposed model from that original deal with a form of incremental training most likely the type of "training" which is subject to child in the first twenty-four months of life. In Figure 31, are shown the trends of the taxonomic factor of the model of Mayor & Plunkett and of our model, respectively during a standard method of training and in the presence of spaced out insertion of the categories (incremental training) calculated after different epochs of training. In all next comparison result, original model trend shown during standard training was taken from [21]. In Figure 32, are

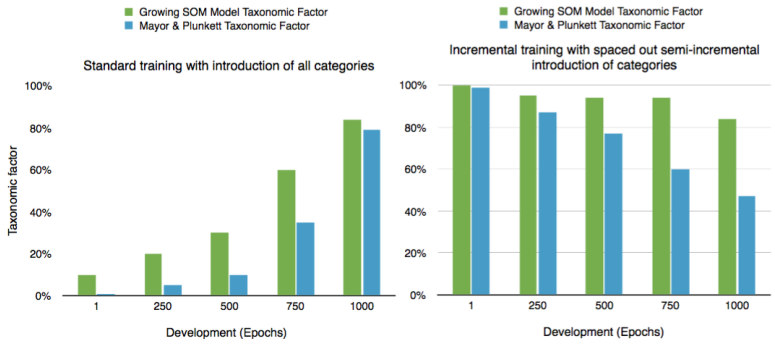

Figure 31: Taxonomic result during maps evolution in standard training (left) and incremental training (right). In blue, taxonomic factor reached by Mayor & Plunkett model, in green by our model.

shown the trends of the quantization error of the model of Mayor & Plunkett and of our model, respectively during a standard method of training and in the presence of spaced out insertion of the categories (incremental training). As shown, the trend recorded during a training standard is similar to the one recorded using SOM static: not only the growing SOM showed a greater increase of taxonomic response thanks to their ability to grow. During the incremental training, the standard model (blue) collapses because the maps suffer of catastrophic interference. The model

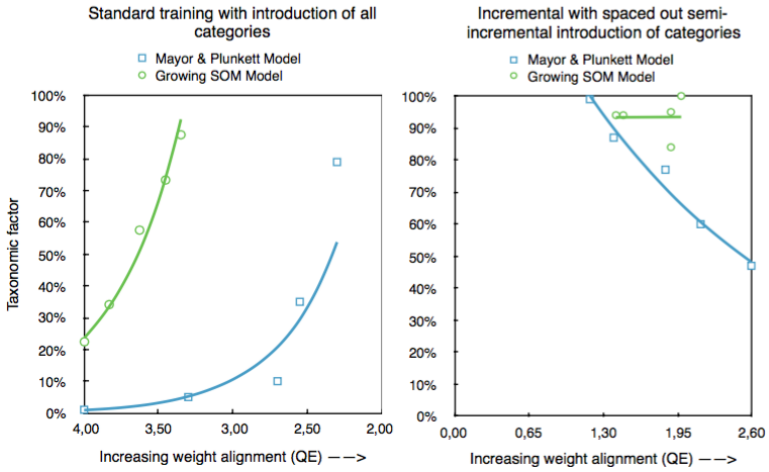

Figure 32: Taxonomic result compared with quantization errors during maps evolution in standard training (left) and incremental training (right). In blue, taxonomic factor reached by Mayor & Plunkett model, in green by our model.

presented growing SOM (green) does not have this problem, as the trend of the taxonomic factor easing curve (remember that it is the index number of learned words). Similar result are shown in Figure 33 related to topological errors trend. We also played a synaptic pruning described by

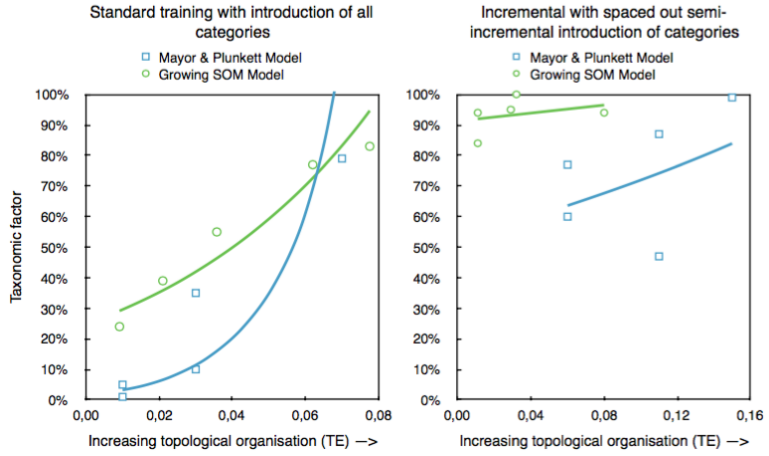

Figure 33: Taxonomic result compared with topological errors during maps evolution in standard training (left) and incremental training (right). In blue, taxonomic factor reached by Mayor & Plunkett model, in green by our model.

the original authors of the model, and also in this circumstance the model with growing SOM has the same performance recorded running the static model, as shown in Figure 34.

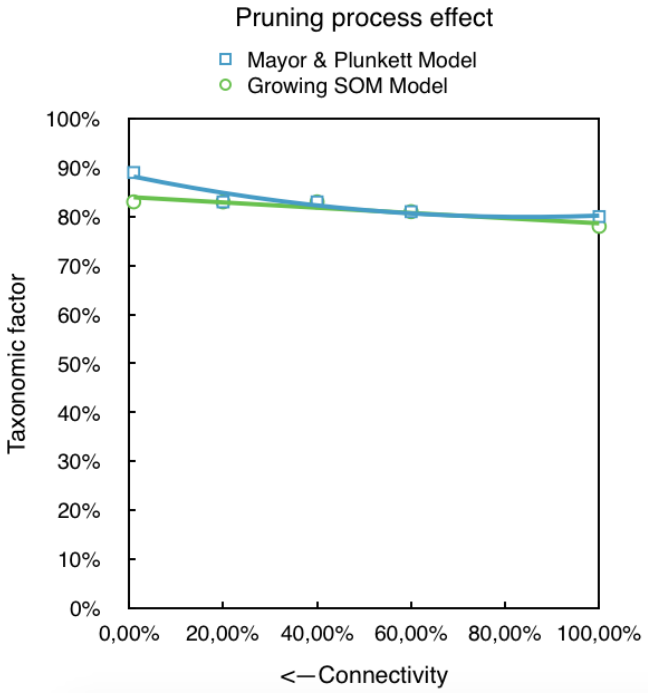

Figure 34: Taxonomic result compared with synaptic connectivity during synaptic pruning phase in standard training. In blue, taxonomic factor reached by Mayor & Plunkett model, in green by our model.

4.6 SUMMARY

In this chapter, we introduced the form of incremental training required to deal with the problem of learning new categories. This has led to the confirmation about the need to introduce new neurons to increase the categorization ability: to solve the problem of choosing a priori the size of the map, we have introduced a form of dynamic expansion which is able to maintain the wired knowledge previously stored, both during Hebbian learning and during the learning maps, responding to the need to categorize exemplars from new categories through a neural growth process that is expected to enroll new units. From the tests conducted on

static maps and maps of different sizes able to grow during training, the Growing SOM are a valid alternative to the standard SOM, as they are able to achieve performance comparable to static maps arbitrarily large with fixed size.

5

NEXT STEPS

In this chapter, we will investigate some possible ideas for development of the current model, with considerations about the maps and techniques through which training can take place.

5.1 FUTURE POSSIBLE EXPANSION

5.1.1 *Final fine-tuning*

Fritzke speaks in [5] about fine-tuning at the end of the expansion of the map, as we said before in subsection 4.3.3. This process is done performing a form of training with decreasing parameters once the stop criterion of maps' growing learning is reached: in our model, we have not achieved this last phase, since the very concept of stop criterion is not contemplated. The question we asked ourselves is: if this step of fine-tuning was necessary, when it would be permissible to insert it? In a real context, a child may be constantly subject to new stimuli: that would make no sense to introduce a form of *crystallization* during a *normal day-of-learning*. However, it is now known that during the hours of sleep the brain is the scene of cognitive processes different from those that are carried out during the day: as the literature grows, a clearer picture is emerging of which memories are processed during sleep and the relevant aspects of sleep in physiology (Rebecca M.C. Spencer, [13]). Different sleep stages are associated with different forms of memory - meaning, sleep is not just as singular memory is not singular. In [13], Rebecca M.C. Spencer & Co show evidence that classroom naps support learning in preschool children by en-

hancing memories acquired earlier in the day compared with equivalent intervals spent awake. This nap benefit is greatest for children who nap habitually, regardless of age. Performance losses when nap-deprived are not recovered during subsequent overnight sleep. Physiological recordings of naps support a role of sleep spindles in memory performance. To investigate whether in-class naps benefit declarative learning in preschool children, Rebecca M.C. Spencer & Co measured changes in performance on a **visuospatial task** over a nap and an equivalent interval of wake. As they said, a visuospatial task was selected for three reasons: first, this task, like other declarative learning tasks, has been shown to engage the hippocampus, and hippocampal-dependent tasks are subject to neural replay during sleep, a possible mechanism underlying sleep-dependent consolidation; second, visuospatial learning has been shown to benefit from overnight sleep in young adults; third, the task, like the game Memory, is appealing to preschool children.

We think a possible evolution of this model provides for the introduction of this stage of fine-tuning in an efficient simulation environment, able to replicate every stage of learning, including the passive one realized during sleep: the stage of fine-tuning mentioned Fritzke might be a good vector to simulate the phase of crystallization of learned memories during the night, including Hebbian synaptic connections, in a totally unsupervised context that is based on what has been learned during the day. In this scenario, it would be interesting to compare the results obtained following the introduction of periodic convergence phases of the maps, with the same results obtained from experiments in which *sleeps* (or *naps*) is not simulated, and see if the *sleeping process* results in quantifiable benefits in the process of word learning.

5.1.2 *Topology and related problems*

An interesting alternative to the expansion technique described in subsection 4.4.3 involves a radically change the topology of the map: if this was toroidal, we could easily add an arbitrary number of units around the unit selected by inserting some "circles of units" on the right and left of

the selected neuron, creating a empty space around it. However, reflecting well on this shape, there would be the same imbalance linked to the most distant units. Conceptually, add rows or/and columns means inserting new units also very distant (topologically) from the one that caused the change. However, a three-dimensional and symmetrical topology would lead the neurons to have an equal number of neighbors and, potentially, also a greater number of neighbors if we think about the torus as a *full* solid. It is difficult to investigate the behavior of a SOM whose neurons are arranged on a three-dimensional surface without performing specific tests, but we know that the neighborhood plays a key role during learning. Introducing a more complex structure would change definitely the way in which the weights are updated: also, conceptually the idea of positioning in three-dimensional space is more natural due to the type of positioning of the neurons in a real brain, moving to a more neurobiologically plausible model. Also in this context, it would be interesting to study the possibly increase of performance of the model in comparison to a model with two-dimensional maps. Investigating the results by taking this path would require a too radical change to the model, and the need to re-evaluate again the SOM ability to categorize the new topology, in addition to a careful study about how the SOM-Toolbox library indexes the units arranged on a symmetrical three-dimensional structure as a toroid.

5.1.3 *Attentional weights*

A problem often faced in models that try to study language learning is related to the attention that is given to the various input parts, or better, the various characteristics that describe input. Models like ALCOVE [12] or Lex [32] provide attentional weights that mediate the importance of each input dimension to build more solid boundaries and better define in semantics (or syntax) of certain categories (words). If in a child the concept of "dog", for instance, exists and is well defined, then it is easy to imagine that the same concept in the same child will have undergone an evolution at the age of ten years: a first approximation of this evolutionary process might be an information enrichment. Thus, numerically speaking, we could say that the concept of dog is first described by a few

representative features, we say X in number, and that over time has been enriched up to, for instance, 3X features. However, the concept of dog is well established probably in every culture that came into contact with the animal: how many and which features describe this concept and others?

This is a one million dollars question, but we can say with certainty that the concept is still described by X features, or even 3X distinct features, in every child age: we say that the number of possible information related to the concept of dog is limited. However, not all this information is available to the child or to a young person or to an adult in the same way: some of this information is removed, as if it never existed. Thus, a shape-bias involved in the categorization of early childhood process, but sooner or later we could image that the previously discarded information will recur and will be stored so as to refine the concept created earlier. This evolution of the input representation can be translated in a change-of-attention given to the various features that describe inputs. Imagine an input featured on 20 different dimensions: a SOM, or similarly a GSOM, defined on a two dimensional grid is displayed as a sheet of neurons. Each is linked to a weight vector with 20 elements. We could interrupt the process of organization on the first three dimensions, and topologically organize a map as if the input were described using only 3 features (for instance, x, y and z, as wide, length and depth). An evolutionary process that incorporates the dimensions spaced over time might be a good approximation of the attention that a child provides to input details that processes. The process of organizing maps could exist until a certain degree of cohesion of the categories identified on the map is reached, and then incorporate the new dimensions and start the organization whereas a new level of depth. One (G) SOM, with attentional weights of any size, may be a good carrier for explaining the variation in the input processing capacity similar to that recorded in the models that explain language learning in adults.

5.1.4 *Expansion criteria*

In the literature, many growing techniques have been proposed: as already said, the idea of B. Fritzke was convenient and effective to imple-

ment in our model because of its intrinsic characteristics. In this contest, simulate the auditory visual cortex could be done using auto-associators, or multi-level networks, or multi-level SOM. There are many possibilities, and many models available in the literature: in the future, it would be interesting to try to take alternative routes that make use of different patterns, while maintaining the psychological coherence, and creating a comprehensive comparison of multiple implementations with different models. For instance, learning by association (Hebbian training) could be used pro-actively to facilitate the formation of well-defined categories on the models. A crucial aspect, which we discussed in part in the paragraph related to the stage of fine-tuning, regards the memory: this is certainly a fundamental element of the entire nervous system. A possible evolution of the model could relate to storing maps (or substitute models of which we talked earlier) in different moments during training phase, in a process that can simulate "removal" of memories, that is, the inability to recognize things learned previously. In this scenario, further studies on the formation of primitive knowledge and secondary knowledge should be made, not to run the risk of turning a simulation model (which aims to simulate a real process to find a solution for a problem) in a resolving model (which aims to find a solution to a real problem regardless of how this is found).

A

APPENDIX - TRIAL RESULTS

This appendix shows the results of the experiment averaged over multiple runs, grouped for readability. At the bottom of the tables there is a description of the test conducted.

Stats/Epochs	Visual Q.Error	Auditory Q.Error	% Learned
All 100 categories after 100 epochs	25.82	11.15	39.25%
All 100 categories after 200 epochs	25.03	10.64	49.08%
All 100 categories after 300 epochs	24.53	10.35	57.08%
All 100 categories after 400 epochs	24.05	10.19	61.67%
All 100 categories after 500 epochs	23.68	10.05	65.50%
All 100 categories after 600 epochs	23.21	9.93	68.50%
All 100 categories after 700 epochs	22.90	9.83	74.83%
All 100 categories after 800 epochs	22.50	9.74	77.17%
All 100 categories after 900 epochs	22.08	9.64	78.58%
All 100 categories after 900 epochs	21.72	9.55	78.58%

Table 13: Visual and auditory maps' quantization error and percentage of learned categories in production test, using static maps both with grid dimension $25 \cdot 25$, during standard SOM training with standard Hebbian training, introduced every 100 epoch without using the old synaptic connections.

Stats/Epochs	Visual Q.Error	Auditory Q.Error	% Learned
All 100 categories after 100 epochs	25.85	11.09	36.67%
All 100 categories after 200 epochs	25.06	10.59	48.50%
All 100 categories after 300 epochs	24.53	10.31	56.08%
All 100 categories after 400 epochs	24.04	10.13	61.42%
All 100 categories after 500 epochs	23.69	9.99	65.08%
All 100 categories after 600 epochs	23.30	9.85	69.25%
All 100 categories after 700 epochs	22.89	9.73	73.83%
All 100 categories after 800 epochs	22.53	9.65	75.33%
All 100 categories after 900 epochs	22.14	9.59	79.08%
All 100 categories after 900 epochs	21.82	9.50	79.08%

Table 14: Visual and auditory maps' quantization error and percentage of learned categories in production test, using static maps both with grid dimension $25 \cdot 25$, during standard SOM training with incremental Hebbian training, introduced every 100 epoch using the same synaptic connections of previous training.

Categories/Epochs	100	200	300	400	500	600	700	800	900	1000
Visual Map Q.Error	14.46	13.62	15.88	17.91	19.01	20.12	20.81	21.68	22.34	22.84
Auditory Map Q.Error	6.68	6.19	6.83	7.60	8.15	8.65	9.00	9.39	9.67	9.93
First 10 categories	100.00%	100.00%	100.00%	98.33%	100.00%	98.33%	96.67%	97.50%	100.00%	98.33%
Next 20 categories (1 – 20)	/	100.00%	99.17%	99.17%	98.33%	98.75%	96.67%	95.42%	96.67%	94.58%
Next 30 categories (1 – 30)	/	/	94.17%	96.67%	96.67%	97.50%	96.67%	94.72%	95.28%	92.50%
Next 40 categories (1 – 40)	/	/	/	88.54%	94.17%	95.83%	95.42%	93.54%	94.38%	92.29%
Next 50 categories (1 – 50)	/	/	/	/	87.00%	92.33%	92.83%	91.33%	92.50%	90.33%
Next 60 categories (1 – 60)	/	/	/	/	/	82.50%	86.39%	85.97%	87.08%	84.31%
Next 70 categories (1 – 70)	/	/	/	/	/	/	77.62%	79.17%	81.43%	78.69%
Next 80 categories (1 – 80)	/	/	/	/	/	/	/	71.67%	73.96%	73.33%
Next 90 categories (1 – 90)	/	/	/	/	/	/	/	/	67.41%	67.69%
Next 100 categories (1 – 100)	/	/	/	/	/	/	/	/	/	61.08%
Tot. correct / Tot. Tested	100.00%	100.00%	94.17%	88.54%	87.00%	82.50%	77.62%	71.67%	67.41%	61.08%

Table 15: Visual and auditory maps' quantization error and percentage of learned categories in production test, using static maps both with grid dimension $25 \cdot 25$, after a spaced out incremental introduction of categories (12 exemplars from 10 new categories every 100 epochs) during SOM training and a single labelling event during Hebbian training (1 exemplar for each `category`).

Categories/Epochs	100	200	300	400	500	600	700	800	900	1000
Visual Map Q.Error	14.46	13.62	15.88	17.91	19.01	20.12	20.81	21.68	22.34	22.84
Auditory Map Q.Error	6.68	6.19	6.83	7.60	8.15	8.65	9.00	9.39	9.67	9.93
First 10 categories	120	120	120	118	120	118	116	117	120	118
Next 20 categories (1 – 20)	/	240	238	238	236	237	232	229	232	227
Next 30 categories (1 – 30)	/	/	339	348	348	351	348	341	343	333
Next 40 categories (1 – 40)	/	/	/	425	452	460	458	449	453	443
Next 50 categories (1 – 50)	/	/	/	/	522	554	557	548	555	542
Next 60 categories (1 – 60)	/	/	/	/	/	594	622	619	627	607
Next 70 categories (1 – 70)	/	/	/	/	/	/	652	665	684	661
Next 80 categories (1 – 80)	/	/	/	/	/	/	/	688	710	704
Next 90 categories (1 – 90)	/	/	/	/	/	/	/	/	728	731
Next 100 categories (1 – 100)	/	/	/	/	/	/	/	/	/	733
Tot. correct / Tot. Tested	120	240	339	425	522	594	652	688	728	733

Table 16: Visual and auditory maps' quantization error and total count of learned exemplars for categories in production test, using static maps both with grid dimension $25 \cdot 25$, after a spaced out incremental introduction of categories (12 exemplars from 10 new categories every 100 epochs) during SOM training and a single labelling event during Hebbian training (1 exemplar for each `category`).

Categories/Epochs	100	200	300	400	500	600	700	800	900	1000
Visual Map Q.Error	12.01	12.09	12.64	13.32	13.19	13.89	14.62	15.46	15.01	30.46
Auditory Map Q.Error	5.44	5.67	5.81	6.02	6.16	6.49	6.71	6.66	6.81	14.08
First 10 categories	95.83%	88.33%	54.17%	34.17%	21.67%	11.67%	7.50%	5.00%	2.50%	2.50%
Next 20 categories (11 – 20)	/	90.83%	72.50%	52.92%	38.33%	23.33%	12.08%	10.42%	5.00%	4.58%
Next 30 categories (21 – 30)	/	/	79.17%	64.72%	50.28%	35.28%	21.11%	15.28%	12.78%	9.72%
Next 40 categories (31 – 40)	/	/	/	72.71%	60.83%	48.12%	35.62%	29.79%	23.12%	20.00%
Next 50 categories (41 – 50)	/	/	/	/	67.83%	57.33%	45.83%	39.67%	30.17%	27.00%
Next 60 categories (51 – 60)	/	/	/	/	/	63.89%	54.03%	47.50%	39.17%	36.11%
Next 70 categories (61 – 70)	/	/	/	/	/	/	59.88%	53.93%	45.83%	41.90%
Next 80 categories (71 – 80)	/	/	/	/	/	/	/	59.69%	51.46%	46.98%
Next 90 categories (81 – 90)	/	/	/	/	/	/	/	/	56.85%	52.87%
Next 100 categories (91 – 100)	/	/	/	/	/	/	/	/	/	47.67%
Tot. correct / Tot. Tested	95.83%	90.83%	79.17%	72.71%	67.83%	63.89%	59.88%	59.69%	56.85%	47.67%

Table 17: Visual and auditory maps' quantization error and percentage of learned categories in production test, using static maps both with grid dimension $25 \cdot 25$, after a spaced out not incremental introduction of categories (12 exemplars from 10 new categories with replacement every 100 epochs) during SOM training and a single labelling event during Hebbian training (1 exemplar for each `category`).

Categories/Epochs	100	200	300	400	500	600	700	800	900	1000
Visual Map Q.Error	12.01	12.09	12.64	13.32	13.19	13.89	14.62	15.46	15.01	30.46
Auditory Map Q.Error	5.44	5.67	5.81	6.02	6.16	6.49	6.71	6.66	6.81	14.08
First 10 categories	115	106	65	41	26	14	9	6	3	3
Next 20 categories (11 – 20)	/	218	174	127	92	56	29	25	12	11
Next 30 categories (21 – 30)	/	/	285	233	181	127	76	55	46	35
Next 40 categories (31 – 40)	/	/	/	349	292	231	171	143	111	96
Next 50 categories (41 – 50)	/	/	/	/	407	344	275	238	181	162
Next 60 categories (51 – 60)	/	/	/	/	/	460	389	342	282	260
Next 70 categories (61 – 70)	/	/	/	/	/	/	503	453	385	352
Next 80 categories (71 – 80)	/	/	/	/	/	/	/	573	494	451
Next 90 categories (81 – 90)	/	/	/	/	/	/	/	/	614	571
Next 100 categories (91 – 100)	/	/	/	/	/	/	/	/	/	572
Tot. correct / Tot. Tested	115	218	285	349	407	460	503	573	614	572

Table 18: Visual and auditory maps' quantization error and total count of learned exemplars for categories in production test, using static maps both with grid dimension $25 \cdot 25$, after a spaced out not incremental introduction of categories (12 exemplars from 10 new categories with replacement every 100 epochs) during SOM training and a single labelling event during Hebbian training (1 exemplar for each `category`).

Categories/Epochs	100	200	300	400	500	600	700	800	900	1000
Visual Map Q.Error	11.80	13.08	14.56	16.73	18.34	19.92	20.44	21.62	22.37	23.52
Auditory Map Q.Error	5.42	5.64	6.32	6.87	7.82	8.31	8.98	9.35	9.61	10.36
First 10 categories	100.00%	97.50%	98.33%	95.00%	98.33%	93.33%	93.33%	89.17%	87.50%	90.00%
Next 20 categories (1 – 20)	/	98.75%	98.33%	95.83%	96.67%	92.92%	92.08%	89.17%	88.75%	89.58%
Next 30 categories (1 – 30)	/	/	98.89%	94.17%	96.11%	89.44%	89.72%	89.17%	88.61%	90.28%
Next 40 categories (1 – 40)	/	/	/	93.96%	95.00%	89.58%	88.96%	88.75%	87.08%	89.38%
Next 50 categories (1 – 50)	/	/	/	/	93.33%	89.17%	89.33%	89.83%	88.67%	89.83%
Next 60 categories (1 – 60)	/	/	/	/	/	87.22%	88.19%	88.89%	89.03%	88.61%
Next 70 categories (1 – 70)	/	/	/	/	/	/	84.88%	85.24%	85.95%	85.60%
Next 80 categories (1 – 80)	/	/	/	/	/	/	/	80.73%	82.08%	81.77%
Next 90 categories (1 – 90)	/	/	/	/	/	/	/	/	78.33%	81.02%
Next 100 categories (1 – 100)	/	/	/	/	/	/	/	/	/	73.00%
Tot. correct / Tot. Tested	100.00%	98.75%	98.89%	93.96%	93.33%	87.22%	84.88%	80.73%	78.33%	73.00%

Table 19: Visual and auditory maps' quantization error and percentage of learned categories in production test, using static maps both with grid dimension $25 \cdot 25$, after a spaced out semi-incremental introduction of categories (12 exemplars from 10 new categories and 4 for old-categories every 100 epochs) during SOM training and a single labelling event during Hebbian training (1 exemplar for each `category`).

Categories/Epochs	100	200	300	400	500	600	700	800	900	1000
Visual Map Q.Error	11.80	13.08	14.56	16.73	18.34	19.92	20.44	21.62	22.37	23.52
Auditory Map Q.Error	5.42	5.64	6.32	6.87	7.82	8.31	8.98	9.35	9.61	10.36
First 10 categories	120	117	118	114	118	112	112	107	105	108
Next 20 categories (1 – 20)	/	237	236	230	232	223	221	214	213	215
Next 30 categories (1 – 30)	/	/	356	339	346	322	323	321	319	325
Next 40 categories (1 – 40)	/	/	/	451	456	430	427	426	418	429
Next 50 categories (1 – 50)	/	/	/	/	560	535	536	539	532	539
Next 60 categories (1 – 60)	/	/	/	/	/	628	635	640	641	638
Next 70 categories (1 – 70)	/	/	/	/	/	/	713	716	722	719
Next 80 categories (1 – 80)	/	/	/	/	/	/	/	775	788	785
Next 90 categories (1 – 90)	/	/	/	/	/	/	/	/	846	875
Next 100 categories (1 – 100)	/	/	/	/	/	/	/	/	/	876
Tot. correct / Tot. Tested	120	237	356	451	560	628	713	775	846	876

Table 20: Visual and auditory maps' quantization error and total count of learned exemplars for categories in production test, using static maps both with grid dimension $25 \cdot 25$, after a spaced out semi-incremental introduction of categories (12 exemplars from 10 new categories and 4 for old-categories every 100 epochs) during SOM training and a single labelling event during Hebbian training (1 exemplar for each `category`).

Categories/Epochs	100	200	300	400	500	600	700	800	900	1000
Visual Map Q.Error	11.78	11.52	11.62	12.39	11.91	11.52	12.27	11.58	12.06	32.82
Auditory Map Q.Error	5.50	5.27	5.59	5.03	5.19	5.29	5.19	5.39	5.26	14.72
First 10 categories	89.17%	79.17%	23.33%	10.00%	5.83%	2.50%	3.33%	3.33%	/	5.00%
Next 20 categories (11 – 20)	/	82.50%	48.75%	26.67%	9.17%	5.00%	4.17%	2.92%	0.42%	4.58%
Next 30 categories (21 – 30)	/	/	60.56%	33.06%	13.06%	5.56%	5.56%	3.89%	1.39%	3.89%
Next 40 categories (31 – 40)	/	/	/	46.04%	26.04%	11.25%	8.12%	3.54%	1.04%	3.33%
Next 50 categories (41 – 50)	/	/	/	/	38.17%	20.83%	13.00%	4.50%	1.33%	3.67%
Next 60 categories (51 – 60)	/	/	/	/	/	31.67%	21.94%	7.36%	2.64%	4.03%
Next 70 categories (61 – 70)	/	/	/	/	/	/	30.71%	15.24%	6.67%	5.48%
Next 80 categories (71 – 80)	/	/	/	/	/	/	/	25.00%	13.44%	10.62%
Next 90 categories (81 – 90)	/	/	/	/	/	/	/	/	22.87%	20.00%
Next 100 categories (91 – 100)	/	/	/	/	/	/	/	/	/	18.17%
Tot. correct / Tot. Tested	89.17%	82.50%	60.56%	46.04%	38.17%	31.67%	30.71%	25.00%	22.87%	18.17%

Table 21: Visual and auditory maps' quantization error and percentage of learned categories in production test, using static maps both with grid dimension $25 \cdot 25$, after a spaced out not incremental introduction of categories (12 exemplars from 10 new categories with replacement every 100 epochs) during SOM training, with $1 - 100$ organization phase learning/neighborhood descending values, every 100 epochs, and a single labelling event during Hebbian training (1 exemplar for each `category`).

Categories/Epochs	100	200	300	400	500	600	700	800	900	1000
Visual Map Q.Error	11.78	11.52	11.62	12.39	11.91	11.52	12.27	11.58	12.06	32.82
Auditory Map Q.Error	5.50	5.27	5.59	5.03	5.19	5.29	5.19	5.39	5.26	14.72
First 10 categories	107	95	28	12	7	3	4	4	/	6
Next 20 categories ($11-20$)	/	198	117	64	22	12	10	7	1	11
Next 30 categories ($21-30$)	/	/	218	119	47	20	20	14	5	14
Next 40 categories ($31-40$)	/	/	/	221	125	54	39	17	5	16
Next 50 categories ($41-50$)	/	/	/	/	229	125	78	27	8	22
Next 60 categories ($51-60$)	/	/	/	/	/	228	158	53	19	29
Next 70 categories ($61-70$)	/	/	/	/	/	/	258	128	56	46
Next 80 categories ($71-80$)	/	/	/	/	/	/	/	240	129	102
Next 90 categories ($81-90$)	/	/	/	/	/	/	/	/	247	216
Next 100 categories ($91-100$)	/	/	/	/	/	/	/	/	/	218
Tot. correct / Tot. Tested	107	198	218	221	229	228	258	240	247	218

Table 22: Visual and auditory maps' quantization error and total count of learned exemplars for categories in production test, using static maps both with grid dimension $25 \cdot 25$, after a spaced out not incremental introduction of categories (12 exemplars from 10 new categories with replacement every 100 epochs) during SOM training, with $1-100$ organization phase learning/neighborhood descending values, every 100 epochs, and a single labelling event during Hebbian training (1 exemplar for each `category`).

Categories/Epochs	100	200	300	400	500	600	700	800	900	1000
Visual Map Q.Error	11.82	12.61	15.24	16.94	18.42	20.20	21.06	22.16	23.42	25.93
Auditory Map Q.Error	5.45	5.60	6.09	6.75	7.60	8.15	8.62	9.02	9.58	10.86
First 10 categories	95.00%	92.50%	77.50%	80.83%	75.00%	70.83%	50.83%	48.33%	50.00%	46.67%
Next 20 categories ($1-20$)	/	92.92%	82.92%	82.08%	75.42%	67.50%	55.42%	46.67%	49.58%	45.42%
Next 30 categories ($1-30$)	/	/	85.00%	82.50%	75.00%	68.06%	59.17%	51.39%	48.06%	46.67%
Next 40 categories ($1-40$)	/	/	/	84.38%	75.00%	64.79%	57.71%	50.62%	45.21%	46.04%
Next 50 categories ($1-50$)	/	/	/	/	77.33%	67.33%	56.67%	49.83%	45.67%	47.17%
Next 60 categories ($1-60$)	/	/	/	/	/	71.11%	59.44%	50.28%	47.78%	47.78%
Next 70 categories ($1-70$)	/	/	/	/	/	/	63.33%	53.57%	46.90%	47.14%
Next 80 categories ($1-80$)	/	/	/	/	/	/	/	57.92%	48.54%	47.50%
Next 90 categories ($1-90$)	/	/	/	/	/	/	/	/	52.96%	52.69%
Next 100 categories ($1-100$)	/	/	/	/	/	/	/	/	/	47.83%
Tot. correct / Tot. Tested	95.00%	92.92%	85.00%	84.38%	77.33%	71.11%	63.33%	57.92%	52.96%	47.83%

Table 23: Visual and auditory maps' quantization error and percentage of learned categories in production test, using static maps both with grid dimension $25 \cdot 25$, after a spaced out semi-incremental introduction of categories (12 exemplars from 10 new categories and 4 for old-categories every 100 epochs) during SOM training, with $1-100$ organization phase learning/neighborhood descending values, every 100 epochs, and a single labelling event during Hebbian training (1 exemplar for each `category`).

Categories/Epochs	100	200	300	400	500	600	700	800	900	1000
Visual Map Q.Error	11.82	12.61	15.24	16.94	18.42	20.20	21.06	22.16	23.42	25.93
Auditory Map Q.Error	5.45	5.60	6.09	6.75	7.60	8.15	8.62	9.02	9.58	10.86
First 10 categories	114	111	93	97	90	85	61	58	60	56
Next 20 categories ($1-20$)	/	223	199	197	181	162	133	112	119	109
Next 30 categories ($1-30$)	/	/	306	297	270	245	213	185	173	168
Next 40 categories ($1-40$)	/	/	/	405	360	311	277	243	217	221
Next 50 categories ($1-50$)	/	/	/	/	464	404	340	299	274	283
Next 60 categories ($1-60$)	/	/	/	/	/	512	428	362	344	344
Next 70 categories ($1-70$)	/	/	/	/	/	/	532	450	394	396
Next 80 categories ($1-80$)	/	/	/	/	/	/	/	556	466	456
Next 90 categories ($1-90$)	/	/	/	/	/	/	/	/	572	569
Next 100 categories ($1-100$)	/	/	/	/	/	/	/	/	/	574
Tot. correct / Tot. Tested	114	223	306	405	464	512	532	556	572	574

Table 24: Visual and auditory maps' quantization error and total count of learned exemplars for categories in production test, using static maps both with grid dimension $25 \cdot 25$, after a spaced out semi-incremental introduction of categories (12 exemplars from 10 new categories and 4 for old-categories every 100 epochs) during SOM training, with $1-100$ organization phase learning/neighborhood descending values, every 100 epochs, and a single labelling event during Hebbian training (1 exemplar for each `category`).

Categories/Epochs	100	200	300	400	500	600	700	800	900	1000
Visual Map Q.Error	10.93	11.09	11.01	11.20	11.41	12.08	12.39	13.04	13.92	17.40
Auditory Map Q.Error	5.13	5.00	5.06	5.17	5.14	5.32	5.44	5.64	5.93	7.36
First 10 categories	100.00%	100.00%	100.00%	100.00%	100.00%	98.33%	100.00%	100.00%	100.00%	100.00%
Next 20 categories ($1-20$)	/	100.00%	100.00%	100.00%	100.00%	99.17%	100.00%	100.00%	100.00%	100.00%
Next 30 categories ($1-30$)	/	/	100.00%	100.00%	100.00%	99.44%	100.00%	100.00%	100.00%	100.00%
Next 40 categories ($1-40$)	/	/	/	100.00%	100.00%	99.17%	99.58%	99.17%	100.00%	99.58%
Next 50 categories ($1-50$)	/	/	/	/	99.83%	98.67%	99.67%	99.00%	99.83%	99.50%
Next 60 categories ($1-60$)	/	/	/	/	/	98.06%	99.17%	98.61%	99.86%	98.61%
Next 70 categories ($1-70$)	/	/	/	/	/	/	99.05%	98.45%	99.64%	98.10%
Next 80 categories ($1-80$)	/	/	/	/	/	/	/	98.23%	98.65%	97.60%
Next 90 categories ($1-90$)	/	/	/	/	/	/	/	/	97.04%	97.50%
Next 100 categories ($1-100$)	/	/	/	/	/	/	/	/	/	88.00%
Tot. correct / Tot. Tested	100.00%	100.00%	100.00%	100.00%	99.83%	98.06%	99.05%	98.23%	97.04%	88.00%

Table 25: Visual and auditory maps' quantization error and percentage of learned categories in production test, using static maps both with grid dimension $50 \cdot 50$, after a spaced out semi-incremental introduction of categories (12 exemplars from 10 new categories and 4 for old-categories every 100 epochs) during SOM training, with $1-100$ organization phase learning/neighborhood descending values, every 100 epochs, and a single labelling event during Hebbian training (1 exemplar for each `category`).

Categories/Epochs	100	200	300	400	500	600	700	800	900	1000
Visual Map Q.Error	10.93	11.09	11.01	11.20	11.41	12.08	12.39	13.04	13.92	17.40
Auditory Map Q.Error	5.13	5.00	5.06	5.17	5.14	5.32	5.44	5.64	5.93	7.36
First 10 categories	120	120	120	120	120	118	120	120	120	120
Next 20 categories $(1-20)$	/	240	240	240	240	238	240	240	240	240
Next 30 categories $(1-30)$	/	/	360	360	360	358	360	360	360	360
Next 40 categories $(1-40)$	/	/	/	480	480	476	478	476	480	478
Next 50 categories $(1-50)$	/	/	/	/	599	592	598	594	599	597
Next 60 categories $(1-60)$	/	/	/	/	/	706	714	710	719	710
Next 70 categories $(1-70)$	/	/	/	/	/	/	832	827	837	824
Next 80 categories $(1-80)$	/	/	/	/	/	/	/	943	947	937
Next 90 categories $(1-90)$	/	/	/	/	/	/	/	/	1048	1053
Next 100 categories $(1-100)$	/	/	/	/	/	/	/	/	/	1056
Tot. correct / Tot. Tested	120	240	360	480	599	706	832	943	1048	1056

Table 26: Visual and auditory maps' quantization error and total count of learned exemplars for categories in production test, using static maps both with grid dimension $50 \cdot 50$, after a spaced out semi-incremental introduction of categories (12 exemplars from 10 new categories and 4 for old-categories every 100 epochs) during SOM training, with $1-100$ organization phase learning/neighborhood descending values, every 100 epochs, and a single labelling event during Hebbian training (1 exemplar for each `category`).

Categories/Epochs	100	200	300	400	500	600	700	800	900	1000
Visual Map Q.Error	24.08	24.16	23.07	22.93	22.31	21.73	21.75	21.83	21.48	22.47
Auditory Map Q.Error	11.01	10.44	10.57	10.55	10.36	10.31	10.35	10.01	10.00	10.22
First 10 categories	77.50%	73.33%	75.83%	80.83%	76.67%	79.17%	70.83%	69.17%	79.17%	78.33%
Next 20 categories ($1-20$)	/	68.75%	71.67%	77.08%	75.83%	78.33%	74.58%	73.75%	80.00%	80.42%
Next 30 categories ($1-30$)	/	/	68.06%	74.44%	73.06%	76.11%	74.17%	76.11%	80.83%	81.94%
Next 40 categories ($1-40$)	/	/	/	68.12%	67.29%	70.83%	70.00%	71.25%	77.29%	78.12%
Next 50 categories ($1-50$)	/	/	/	/	65.83%	69.17%	68.33%	71.00%	76.50%	77.17%
Next 60 categories ($1-60$)	/	/	/	/	/	65.00%	65.42%	68.75%	74.72%	74.58%
Next 70 categories ($1-70$)	/	/	/	/	/	/	61.55%	67.38%	72.38%	71.07%
Next 80 categories ($1-80$)	/	/	/	/	/	/	/	65.10%	70.94%	68.85%
Next 90 categories ($1-90$)	/	/	/	/	/	/	/	/	68.70%	69.26%
Next 100 categories ($1-100$)	/	/	/	/	/	/	/	/	/	63.75%
Tot. correct / Tot. Tested	77.50%	68.75%	68.06%	68.12%	65.83%	65.00%	61.55%	65.10%	68.70%	63.75%

Table 27: Visual and auditory maps' quantization error and percentage of learned categories in production test, using growing maps both with starting grid dimension $25 \cdot 25$, with expansion in maximum variance direction (first method), after a spaced out semi-incremental introduction of categories (12 exemplars from 10 new categories and 4 for old-categories every 100 epochs) during SOM training, with $1-100$ organization phase learning/neighborhood descending values, every 100 epochs, and a single labelling event during Hebbian training (1 exemplar for each `category`).

Categories/Epochs	100	200	300	400	500	600	700	800	900	1000
Visual Map Q.Error	24.08	24.16	23.07	22.93	22.31	21.73	21.75	21.83	21.48	22.47
Auditory Map Q.Error	11.01	10.44	10.57	10.55	10.36	10.31	10.35	10.01	10.00	10.22
First 10 categories	93	88	91	97	92	95	85	83	95	94
Next 20 categories $(1-20)$	/	165	172	185	182	188	179	177	192	193
Next 30 categories $(1-30)$	/	/	245	268	263	274	267	274	291	295
Next 40 categories $(1-40)$	/	/	/	327	323	340	336	342	371	375
Next 50 categories $(1-50)$	/	/	/	/	395	415	410	426	459	463
Next 60 categories $(1-60)$	/	/	/	/	/	468	471	495	538	537
Next 70 categories $(1-70)$	/	/	/	/	/	/	517	566	608	597
Next 80 categories $(1-80)$	/	/	/	/	/	/	/	625	681	661
Next 90 categories $(1-90)$	/	/	/	/	/	/	/	/	742	748
Next 100 categories $(1-100)$	/	/	/	/	/	/	/	/	/	765
Tot. correct / Tot. Tested	93	165	245	327	395	468	517	625	742	765

Table 28: Visual and auditory maps' quantization error and total count of learned exemplars for categories in production test, using growing maps both with starting grid dimension $25 \cdot 25$, with expansion in maximum variance direction (first method), after a spaced out semi-incremental introduction of categories (12 exemplars from 10 new categories and 4 for old-categories every 100 epochs) during SOM training, with $1-100$ organization phase learning/neighborhood descending values, every 100 epochs, and a single labelling event during Hebbian training (1 exemplar for each `category`).

Categories/Epochs	100	200	300	400	500	600	700	800	900	1000
Visual Map Q.Error	19.79	22.23	17.91	16.14	14.74	14.80	15.26	14.99	15.72	18.77
Auditory Map Q.Error	7.67	7.39	8.32	7.36	6.56	6.63	6.58	6.26	6.37	7.70
First 10 categories	100.00%	95.83%	94.17%	98.33%	96.67%	99.17%	98.33%	93.33%	99.17%	98.33%
Next 20 categories ($1-20$)	/	92.08%	96.25%	95.00%	90.83%	95.42%	96.25%	92.50%	95.42%	95.00%
Next 30 categories ($1-30$)	/	/	95.56%	94.72%	91.67%	96.39%	97.22%	93.06%	95.83%	94.44%
Next 40 categories ($1-40$)	/	/	/	95.42%	92.71%	96.46%	97.50%	94.79%	96.46%	95.42%
Next 50 categories ($1-50$)	/	/	/	/	93.50%	96.17%	96.50%	95.83%	96.50%	96.00%
Next 60 categories ($1-60$)	/	/	/	/	/	95.97%	95.56%	95.14%	95.42%	95.00%
Next 70 categories ($1-70$)	/	/	/	/	/	/	94.05%	94.17%	93.81%	94.17%
Next 80 categories ($1-80$)	/	/	/	/	/	/	/	94.69%	94.58%	94.27%
Next 90 categories ($1-90$)	/	/	/	/	/	/	/	/	93.15%	94.26%
Next 100 categories ($1-100$)	/	/	/	/	/	/	/	/	/	84.83%
Tot. correct / Tot. Tested	100.00%	92.08%	95.56%	95.42%	93.50%	95.97%	94.05%	94.69%	93.15%	84.83%

Table 29: Visual and auditory maps' quantization error and percentage of learned categories in production test, using growing maps both with starting grid dimension $25 \cdot 25$, with expansion in all direction (second winning method), after a spaced out semi-incremental introduction of categories (12 exemplars from 10 new categories and 4 for old-categories every 100 epochs) during SOM training, with $1-100$ organization phase learning/neighborhood descending values, every 100 epochs, and a single labelling event during Hebbian training (1 exemplar for each `category`).

Categories/Epochs	100	200	300	400	500	600	700	800	900	1000
Visual Map Q.Error	19.79	22.23	17.91	16.14	14.74	14.80	15.26	14.99	15.72	18.77
Auditory Map Q.Error	7.67	7.39	8.32	7.36	6.56	6.63	6.58	6.26	6.37	7.70
First 10 categories	120	115	113	118	116	119	118	112	119	118
Next 20 categories ($1-20$)	/	221	231	228	218	229	231	222	229	228
Next 30 categories ($1-30$)	/	/	344	341	330	347	350	335	345	340
Next 40 categories ($1-40$)	/	/	/	458	445	463	468	455	463	458
Next 50 categories ($1-50$)	/	/	/	/	561	577	579	575	579	576
Next 60 categories ($1-60$)	/	/	/	/	/	691	688	685	687	684
Next 70 categories ($1-70$)	/	/	/	/	/	/	790	791	788	791
Next 80 categories ($1-80$)	/	/	/	/	/	/	/	909	908	905
Next 90 categories ($1-90$)	/	/	/	/	/	/	/	/	1006	1018
Next 100 categories ($1-100$)	/	/	/	/	/	/	/	/	/	1018
Tot. correct / Tot. Tested	120	221	344	458	561	691	790	909	1006	1018

Table 30: Visual and auditory maps' quantization error and total count of learned exemplars for categories in production test, using growing maps both with starting grid dimension $25 \cdot 25$, with expansion in all direction (second winning method), after a spaced out semi-incremental introduction of categories (12 exemplars from 10 new categories and 4 for old-categories every 100 epochs) during SOM training, with $1-100$ organization phase learning/neighborhood descending values, every 100 epochs, and a single labelling event during Hebbian training (1 exemplar for each `category`).

BIBLIOGRAPHY

[1] DA Allport. Distributed memory, modular subsystems and dysphasia. *Current perspectives in dysphasia*, pages 32–60, 1985. (cit. a p. 40)

[2] Lawrence W Barsalou. Ad hoc categories. *Memory & cognition*, 11(3): 211–227, 1983. (cit. a p. 85)

[3] Gail A Carpenter and Stephen Grossberg. Art 2: Self-organization of stable category recognition codes for analog input patterns. *Applied optics*, 26(23):4919–4930, 1987. (cit. a p. 85)

[4] Yves Chauvin. *Symbol acquisition in humans and neural (PDP) networks*. PhD thesis, 1989. (cit. a p. 82)

[5] Bernd Fritzke. Growing grid—a self-organizing network with constant neighborhood range and adaptation strength. *Neural Processing Letters*, 2(5):9–13, 1995. (cit. a p. 67, 87, 88, 89, 90, and 107)

[6] Simon Haykin. *Neural Networks: A Comprehensive Foundation*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 2nd edition, 1998. ISBN 0132733501. (cit. a p. 5)

[7] D Hebb. The organization of behaviour: A neuropsychological theory. 1949. (cit. a p. 6 and 40)

[8] John J. Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the national academy of sciences*, 79(8):2554–2558, 1982. (cit. a p. 26)

[9] Peter R Huttenlocher. Neural plasticity: The effects of environment on the development of the cerebral cortex. 2002. (cit. a p. 41)

[10] Andrew G Knapp and James A Anderson. Theory of categorization based on distributed memory storage. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 10(4):616, 1984. (cit. a p. 82)

[11] Teuvo Kohonen. Self-organizing maps, vol. 30 of springer series in information sciences, 2001. (cit. a p. 5)

[12] John K Kruschke. Alcove: an exemplar-based connectionist model of category learning. *Psychological review*, 99(1):22, 1992. (cit. a p. 83 and 109)

[13] Laura Kurdziel, Kasey Duclos, and Rebecca MC Spencer. Sleep spindles in midday naps enhance learning in preschool children. *Proceedings of the National Academy of Sciences*, 110(43):17267–17272, 2013. (cit. a p. 107)

[14] Ping Li, Igor Farkas, and Brian MacWhinney. Early lexical development in a self-organizing neural network. *Neural networks*, 17(8): 1345–1362, 2004. (cit. a p. 85)

[15] Ping Li, Xiaowei Zhao, and Brian Mac Whinney. Dynamic self-organization and early lexical development in children. *Cognitive science*, 31(4):581–612, 2007. (cit. a p. 85)

[16] Bradley C Love, Douglas L Medin, and Todd M Gureckis. Sustain: a network model of category learning. *Psychological review*, 111(2):309, 2004. (cit. a p. 86)

[17] Ellen M. Markman. Constraints children place on word meanings. *Cognitive Science*, 14:57–77, 1990. (cit. a p. 24)

[18] Ellen M. Markman. Constraints on word learning: Speculations about their nature, origins, and domain specificity. 1992. (cit. a p. 26 and 27)

[19] Ellen M Markman, Judith L Wasow, and Mikkel B Hansen. Use of the mutual exclusivity assumption by young word learners. *Cognitive psychology*, 47(3):241–275, 2003. (cit. a p. 28)

[20] Lori Markson, Paul Bloom, et al. Evidence against a dedicated system for word learning in children. *Nature*, 385(6619):813–815, 1997. (cit. a p. 83)

[21] Julien Mayor and Kim Plunkett. A neurocomputational account of taxonomic responding and fast mapping in early word learning. *Psychological review*, 117(1):1, 2010. (cit. a p. 23, 24, 25, 27, 29, 32, 41, 42, 45, 67, 68, 79, 83, 84, 85, and 103)

[22] William E Merriman. Lexical processing. *The emergence of language*, page 331, 1999. (cit. a p. 83)

[23] Gregory Murphy. *The big book of concepts*. MIT press, 2004. (cit. a p. 68)

[24] R. Murphy and R. Honey. *The Wiley Handbook on the Cognitive Neuroscience of Learning*. Wiley, 2015. ISBN 9781118650943. (cit. a p. 24, 25, 26, and 82)

[25] Robert M Nosofsky. Attention, similarity, and the identification–categorization relationship. *Journal of experimental psychology: General*, 115(1):39, 1986. (cit. a p. 83)

[26] Andrew P Papliński. Incremental self-organizing map (isom) in categorization of visual objects. In *Neural Information Processing*, pages 125–132. Springer, 2012. (cit. a p. 25, 68, and 87)

[27] Kim Plunkett, Chris Sinha, Martin F Møller, and Ole Strandsby. Symbol grounding or the emergence of symbols? vocabulary growth in children and a connectionist net. *Connection Science*, 4(3-4):293–312, 1992. (cit. a p. 27 and 82)

[28] Michael I Posner. An informational approach to thinking. Technical report, DTIC Document, 1962. (cit. a p. 25 and 29)

[29] Michael I Posner. Uncertainty as a predictor of similarity in the study of generalization. *Journal of experimental psychology*, 68(2):113, 1964. (cit. a p. 25, 29, and 82)

[30] Michael I Posner and Steven W Keele. On the genesis of abstract ideas. *Journal of experimental psychology*, 77(3p1):353, 1968. (cit. a p. 25 and 29)

[31] Michael I Posner, Ralph Goldsmith, and Kenneth E Welton Jr. Perceived distance and the classification of distorted patterns. *Journal of experimental psychology*, 73(1):28, 1967. (cit. a p. 25, 28, 29, and 82)

[32] Terry Regier. The emergence of words: Attentional learning in form and meaning. *Cognitive science*, 29(6):819–865, 2005. (cit. a p. 82 and 109)

[33] F. Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, pages 65–386, 1958. (cit. a p. 5)

[34] Philippe G Schyns. A modular neural network model of concept acquisition. *Cognitive Science*, 15(4):461–508, 1991. (cit. a p. 85)

[35] Alessandro Treves and Edmund T Rolls. Computational analysis of the role of the hippocampus in memory. *Hippocampus*, 4(3):374–391, 1994. (cit. a p. 25)

[36] R. Waxman Sandra and B. Markow Dana. Words as invitations to form categories: Evidence from 12-to 13-month-old infants. *Cognitive psychology*, 29(3):257–302, 1995. (cit. a p. 26)

[37] Katherine S White and James L Morgan. Sub-segmental detail in early lexical representations. *Journal of Memory and Language*, 59(1): 114–132, 2008. (cit. a p. 25)

[38] Wikipedia. Donald o. hebb — Wikipedia, the free encyclopedia, 2016. URL `https://en.wikipedia.org/wiki/Donald_O._Hebb`. [Online; accessed 18-January-2016, modified 10-January-2016]. (cit. a p. 6)

[39] Junlin Zhou and Yan Fu. Clustering high-dimensional data using growing som. In *Advances in Neural Networks–ISNN 2005*, pages 63–68. Springer, 2005. (cit. a p. 87)